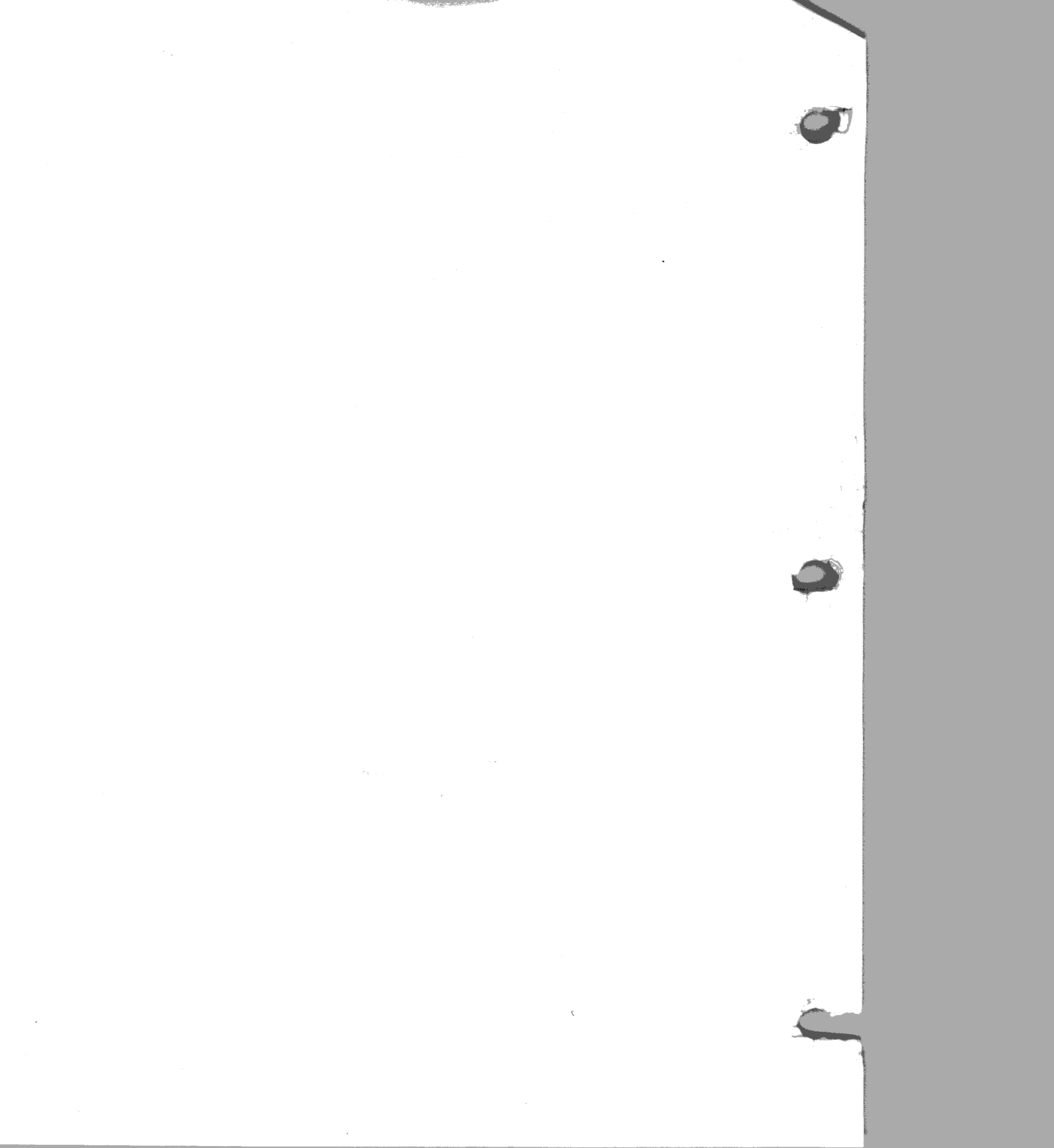


AA-Y506B-TE



# **Guide to VAX/VMS Disk and Magnetic Tape Operations**

Order Number: AI-Y506B-TE

**April 1986**

This guide describes formats and procedures for using disk and magnetic tapes on Version 4.4 of the VAX/VMS operating system. The tasks described in this guide are oriented toward the use of private media.

**Revision/Update Information:**

This revised manual supersedes *Guide to VAX/VMS Disk and Magnetic Tape Operations*, Version 4.0.

**Software Version:**

VAX/VMS Version 4.4

**digital equipment corporation  
maynard, massachusetts**

---

**April 1986**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

---

Copyright ©1986 by Digital Equipment Corporation  
All Rights Reserved

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem-10	PDP	VT
DECSYSTEM-20	PDT	
DECUS	RSTS	
DECwriter	RSX	

**digital**

ZK-2833

---

**HOW TO ORDER ADDITIONAL DOCUMENTATION  
DIRECT MAIL ORDERS**

**USA & PUERTO RICO\***

Digital Equipment Corporation  
P.O. Box CS2008  
Nashua, New Hampshire  
03061

**CANADA**

Digital Equipment  
of Canada Ltd.  
100 Herzberg Road  
Kanata, Ontario K2K 2A6  
Attn: Direct Order Desk

**INTERNATIONAL**

Digital Equipment Corporation  
PSG Business Manager  
c/o Digital's local subsidiary  
or approved distributor

In Continental USA and Puerto Rico call 800-258-1710.

In New Hampshire, Alaska, and Hawaii call 603-884-6660.

In Canada call 800-267-6215.

\* Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminister, Massachusetts 01473.

---

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by T<sub>E</sub>X, the typesetting system developed by Donald E. Knuth at Stanford University. T<sub>E</sub>X is a registered trademark of the American Mathematical Society.

---

# Contents

---

PREFACE

xiii

---

## PART I DISK AND MAGNETIC TAPE CONCEPTS

---

### CHAPTER 1 INTRODUCTION TO DISK AND MAGNETIC TAPE MEDIA 1-1

1.1	THE HISTORY OF FILE STORAGE MEDIA	1-1
1.2	BASIC DISK CONCEPTS	1-2
1.3	BASIC MAGNETIC TAPE CONCEPTS	1-7
1.4	HOW TO IDENTIFY ROUTINE OPERATIONS	1-9
1.5	USING COMMAND PROCEDURES TO PERFORM ROUTINE OPERATIONS	1-9

---

### CHAPTER 2 DISK AND MAGNETIC TAPE PROTECTION 2-1

2.1	DATA PROTECTION	2-1
2.1.1	UIC-Based Protection	2-2
2.1.2	ACL-Based Protection	2-4
2.1.3	VAX/VMS ANSI-Labeled Magnetic Tape Accessibility Protection	2-5
2.2	VOLUME-LEVEL PROTECTION	2-6
2.2.1	Disk Volume Protection	2-6
2.2.2	Magnetic Tape Volume Protection	2-7
2.2.2.1	Protection for VAX/VMS Magnetic Tapes •	2-7
2.2.2.2	Protection for Interchange Environments •	2-8

iii

<b>2.3</b>	<b>FILE-LEVEL PROTECTION</b>	<b>2-8</b>
2.3.1	Disk File Protection	2-9
2.3.2	Directory File Protection	2-11
2.3.3	Magnetic Tape File Protection	2-13

## PART II DISK AND MAGNETIC TAPE OPERATIONS

### CHAPTER 3 PREPARING DISK AND MAGNETIC TAPE VOLUMES FOR PRIVATE USE 3-1

<b>3.1</b>	<b>SETTING UP A PRIVATE VOLUME</b>	<b>3-1</b>
<b>3.2</b>	<b>ALLOCATING A DRIVE TO YOUR PROCESS</b>	<b>3-2</b>
3.2.1	Allocating a Disk Drive	3-2
3.2.2	Allocating a Magnetic Tape Drive	3-3
<b>3.3</b>	<b>INITIALIZING A VOLUME</b>	<b>3-4</b>
3.3.1	Initializing a Disk Volume	3-6
3.3.2	Initializing a Magnetic Tape Volume	3-6
<b>3.4</b>	<b>MOUNTING A VOLUME</b>	<b>3-8</b>
3.4.1	Mounting a Disk Volume	3-10
3.4.2	Mounting a Disk Volume Set	3-11
3.4.2.1	Creating a Disk Volume Set from New Volumes •	3-12
3.4.2.2	Creating a Disk Volume Set from an Existing Volume •	3-13
3.4.2.3	Adding Volumes to a Disk Volume Set •	3-14
3.4.3	Mounting a Magnetic Tape Volume	3-15
3.4.3.1	Mounting an ANSI-Labeled Volume •	3-15
3.4.3.2	Using MOUNT Command Qualifiers •	3-16
3.4.4	Mounting a Magnetic Tape Volume Set	3-19
3.4.4.1	Creating a Magnetic Tape Volume Set •	3-20
3.4.4.2	Mounting Continuation Volumes in a Volume Set •	3-21
<b>3.5</b>	<b>DISMOUNTING A VOLUME</b>	<b>3-23</b>

<b>3.6</b>	<b>DEALLOCATING DRIVES</b>	<b>3-25</b>
------------	----------------------------	-------------

<b>3.7</b>	<b>USING COMMAND PROCEDURES TO SET UP DISKS AND MAGNETIC TAPES</b>	<b>3-26</b>
3.7.1	Designing Command Procedures to Set Up Disk Volumes	3-27
3.7.2	Designing Command Procedures to Set up Magnetic Tape Volumes	3-28

---

## **CHAPTER 4 MANIPULATING FILES ON DISKS AND MAGNETIC TAPES 4-1**

---

<b>4.1</b>	<b>USING DCL TO MANIPULATE FILES</b>	<b>4-1</b>
------------	--------------------------------------	------------

<b>4.2</b>	<b>USING DCL TO RETRIEVE FILE INFORMATION</b>	<b>4-2</b>
4.2.1	Retrieving Directory Information	4-3
4.2.2	Retrieving Device Information	4-4
4.2.3	Retrieving Magnetic Tape Device Information	4-7
4.2.4	Retrieving Disk File Protection Information	4-8
4.2.5	Retrieving Disk Quota Information	4-9

<b>4.3</b>	<b>USING DCL TO MODIFY FILE CHARACTERISTICS</b>	<b>4-10</b>
4.3.1	Modifying ACL Characteristics	4-10
4.3.2	Modifying Directory Characteristics	4-11
4.3.3	Modifying Disk File Characteristics	4-12
4.3.4	Modifying Magnetic Tape Device Characteristics	4-12
4.3.5	Modifying File Protection Characteristics	4-13
4.3.6	Modifying Volume Characteristics	4-15

<b>4.4</b>	<b>USING DCL TO ACCESS FILES</b>	<b>4-15</b>
4.4.1	Accessing Disk Files for Read and Write Operations	4-16
4.4.1.1	Reading Files from a Disk Volume • 4-16	
4.4.1.2	Writing Files to a Disk Volume • 4-17	
4.4.1.3	Writing Files from Disk Volumes to Magnetic Tape Volumes • 4-18	

<b>4.4.2</b>	<b>Accessing Magnetic Tape Files for Read and Write Operations</b>	<b>4-19</b>
4.4.2.1	Locating ANSI-Labeled Magnetic Tape Files for READ or WRITE Access • 4-20	
4.4.2.2	Reading Files on Magnetic Tape Volumes • 4-22	
4.4.2.3	Writing to Files on Magnetic Tape Volumes • 4-23	

<b>4.5</b>	<b>COMMAND PROCEDURES FOR ACCESSING FOREIGN VOLUMES</b>	<b>4-24</b>
------------	---	-------------

---

## **CHAPTER 5 TRANSFERRING DISK AND MAGNETIC TAPE INFORMATION 5-1**

---

<b>5.1</b>	<b>TRANSFERRING INFORMATION WITHIN AND ACROSS OPERATING SYSTEMS</b>	<b>5-1</b>
------------	---	------------

<b>5.2</b>	<b>USING THE COPY COMMAND TO TRANSFER INFORMATION</b>	<b>5-2</b>
5.2.1	Copying Files from Disk Volumes	5-2
5.2.2	Copying Files from Magnetic Tape Volumes	5-3
5.2.3	Copying Files to and from Non-File-Structured Volumes	5-6
5.2.3.1	Copying Files to a Non-File-Structured Volume • 5-6	
5.2.3.2	Copying Files from a Non-File-Structured Volume • 5-7	

<b>5.3</b>	<b>USING THE CONVERT UTILITY TO TRANSFER INFORMATION</b>	<b>5-8</b>
------------	--	------------

<b>5.4</b>	<b>USING THE EXCHANGE UTILITY TO TRANSFER INFORMATION</b>	<b>5-10</b>
5.4.1	Invoking and Terminating the Exchange Utility	5-11
5.4.2	Using EXCHANGE at DCL Command Level	5-12

<b>5.5</b>	<b>USING COMMAND PROCEDURES TO TRANSFER INFORMATION</b>	<b>5-12</b>
5.5.1	Using a Command Procedure to Copy Files	5-13
5.5.2	Using a Command Procedure to Exchange Information	5-15

<b>CHAPTER 6</b>	<b>MAINTAINING DATA INTEGRITY ON DISK AND MAGNETIC TAPE VOLUMES</b>	<b>6-1</b>
<b>6.1</b>	<b>USING BACKUP TO MAINTAIN DATA INTEGRITY</b>	<b>6-2</b>
<b>6.2</b>	<b>CHOOSING BACKUP OPERATIONS</b>	<b>6-3</b>
6.2.1	Deciding <i>What</i> to Back Up	6-3
6.2.2	Deciding <i>How</i> to Back Up Files or Volumes	6-4
6.2.2.1	Copying Files • 6-4	
6.2.2.2	Saving Files • 6-5	
6.2.2.3	Restoring Files • 6-5	
6.2.2.4	Comparing Files • 6-5	
6.2.2.5	Listing Files • 6-5	
6.2.3	Invoking the VAX/VMS Backup Utility	6-6
<b>6.3</b>	<b>USING BACKUP MEDIA</b>	<b>6-6</b>
6.3.1	Using BACKUP with Magnetic Tape	6-7
6.3.2	Using BACKUP with Files—11 Disk Volumes	6-7
6.3.3	Using BACKUP with Sequential Disk Volumes	6-8
6.3.4	Remote Files—11 Save Sets	6-9
6.3.5	Rotating Backup Sets	6-9
<b>6.4</b>	<b>PERFORMING BACKUP TASKS</b>	<b>6-10</b>
6.4.1	Copying Disk Files, Directories, and Volumes	6-11
6.4.1.1	Copying Single Files with BACKUP • 6-11	
6.4.1.2	Copying Multiple Files with BACKUP • 6-11	
6.4.1.3	Copying an Entire Directory Tree with BACKUP • 6-12	
6.4.1.4	Copying a Disk Volume to a Disk with BACKUP • 6-12	
6.4.1.5	Copying a Disk Volume Set to a Disk with BACKUP • 6-13	
6.4.2	Saving Disk Files, Directories, and Volumes to Magnetic Tape	6-13
6.4.2.1	Saving Disk Files to Magnetic Tape • 6-14	
6.4.2.2	Saving Directories to Magnetic Tape • 6-14	
6.4.2.3	Saving a Directory Tree to Magnetic Tape • 6-15	
6.4.2.4	Saving Disk Volumes to Magnetic Tape • 6-15	
6.4.2.5	Saving Unstructured Disk Volumes to Magnetic Tape • 6-15	
6.4.3	Saving Disk Files, Directories, and Volumes to Disks	6-16
6.4.3.1	Writing Save Sets on File-Structured Disk Volumes • 6-16	
6.4.3.2	Writing Save Sets on Sequential Disk Volumes • 6-16	
6.4.3.3	Writing Multivolume Sequential Disk Save Sets • 6-17	

<b>6.4.4</b>	<b>Backing Up Full Volumes and Volume Sets</b>	<b>6-18</b>
6.4.4.1	Backing Up a Disk Volume Set When Drives are Limited • 6-19	
<b>6.4.5</b>	<b>Performing Selective Backups</b>	<b>6-20</b>
<b>6.4.6</b>	<b>Protecting a BACKUP Save Set</b>	<b>6-22</b>
<b>6.4.7</b>	<b>Restoring Disk Files, Directories, and Volumes from Save Sets</b>	<b>6-23</b>
6.4.7.1	Restoring Disk Files from Save Sets on Magnetic Tape • 6-23	
6.4.7.2	Restoring Disk Files from Save Sets on Sequential Disk • 6-24	
6.4.7.3	Restoring Disk Files from Save Sets on Multiple Volumes • 6-25	
<b>6.4.8</b>	<b>Restoring Entire Disk Volumes</b>	<b>6-25</b>
6.4.8.1	Changing Volume Initialization Parameters Before Restoring • 6-26	
<b>6.4.9</b>	<b>Listing the Contents of a BACKUP Save Set</b>	<b>6-26</b>
<b>6.4.10</b>	<b>Comparing Files with BACKUP</b>	<b>6-27</b>
<hr/>		
<b>6.5</b>	<b>USING BACKUP JOURNAL FILES</b>	<b>6-28</b>
<hr/>		
<b>6.6</b>	<b>USING COMMAND PROCEDURES TO PERFORM BACKUP OPERATIONS</b>	<b>6-30</b>

---

## **CHAPTER 7 HANDLING OPERATOR-ORIENTED DISK AND MAGNETIC TAPE FUNCTIONS**

<b>7.1</b>	<b>USING THE OPERATOR TERMINAL TO HANDLE USER REQUESTS</b>	<b>7-1</b>
<hr/>		
<b>7.2</b>	<b>HANDLING USER REQUESTS FOR MOUNTING VOLUMES</b>	<b>7-3</b>
7.2.1	Requests for Mounting Disks and Single Magnetic Tape Volumes	7-4
7.2.2	Requests for Mounting Magnetic Tape Volume Sets	7-5
7.2.2.1	Mounting Volume Sets with Automatic Switching • 7-5	
7.2.2.2	Mounting Volume Sets without Automatic Switching • 7-6	
<hr/>		
<b>7.3</b>	<b>HANDLING REQUESTS FROM THE BACKUP UTILITY</b>	<b>7-8</b>
7.3.1	Writing to a Save Set	7-8
7.3.2	Reading from a Save Set	7-9

7.3.3	Recovering from an Error	7-9
7.4	HANDLING MOUNT VERIFICATION TASKS	7-11
7.4.1	Mount Verification for Offline Devices	7-11
7.4.2	Mount Verification for Write-Locked Devices	7-13
7.4.3	Canceling Mount Verification	7-14
7.4.4	Dismounting a Volume to Abort Mount Verification	7-14
APPENDIX A VAX/VMS DISK FILES AND VOLUMES		A-1
A.1	FILES-11 DISK STRUCTURE	A-1
A.1.1	Index File	A-2
A.1.2	Storage Bit Map File	A-2
A.1.3	Bad Block File	A-3
A.1.4	Master File Directory	A-3
A.1.5	Core Image File	A-3
A.1.6	Volume Set List File	A-3
A.1.7	Continuation File	A-3
A.1.8	Backup Log File	A-4
A.1.9	Pending Bad Block Log File	A-4
A.1.10	Files-11 On-Disk Structure Level 1 Versus Structure Level 2	A-4
APPENDIX B VAX/VMS ANSI-LABELED MAGNETIC TAPE		B-1
B.1	LOGICAL FORMAT OF ANSI-LABELED VOLUMES	B-1
B.2	VAX/VMS MAGNETIC TAPE ANCILLARY CONTROL PROCESS (MTAACP)	B-1
B.3	BASIC COMPONENTS OF THE VAX/VMS ANSI-LABELED FORMAT	B-2
B.3.1	Beginning-of-Tape and End-of-Tape Markers	B-2
B.3.2	Tape Marks	B-2
B.3.3	Labels	B-4
B.4	VOLUME AND FILE CONFIGURATIONS	B-4

B.4.1	Single File/Single Volume Configuration	B-6
B.4.2	Single File/Multivolume Configuration	B-7
B.4.3	Multifile/Single Volume Configuration	B-8
B.4.4	Multifile/Multivolume Configuration	B-9
<hr/>		
B.5	VOLUME LABELS	B-9
B.5.1	VOL1 Label	B-9
B.5.1.1	Volume Identifier Field • B-10	
B.5.1.2	Accessibility Field • B-10	
B.5.1.3	Implementation Identifier Field • B-10	
B.5.1.4	Owner Identifier Field • B-10	
B.5.2	VOL2 Label	B-10
B.5.2.1	Volume-Owner Field • B-11	
<hr/>		
B.6	HEADER LABELS	B-11
B.6.1	HDR1	B-11
B.6.1.1	File Identifier Field • B-12	
B.6.1.2	File-Set Identifier Field • B-13	
B.6.1.3	File Section Number and File Sequence Number Fields • B-13	
B.6.1.4	Generation Number and Generation-Version Number Fields • B-14	
B.6.1.5	Creation Date and Expiration Date Fields • B-14	
B.6.1.6	Accessibility Field • B-15	
B.6.1.7	Implementation Identifier Field • B-15	
B.6.2	HDR2 LABEL	B-15
B.6.2.1	Record Format Field • B-15	
B.6.2.2	Block Length Field • B-17	
B.6.2.3	Record Length Field • B-17	
B.6.2.4	Implementation-Dependent Field • B-18	
B.6.2.5	Buffer-Offset Length Field • B-18	
B.6.3	HDR3 LABEL	B-19
B.6.4	HDR4 Label	B-19
<hr/>		
B.7	TRAILER LABELS	B-19

---

## INDEX

---

### FIGURES

1-1	File Extents _____	1-3
1-2	Files-11 On-Disk Structure Hierarchy _____	1-4
1-3	Tracks and Cylinders _____	1-5
1-4	Interrecord Gaps _____	1-8
2-1	Illustrating User Categories with a UIC of [100,100] _____	2-3
B-1	Basic Layout of a VAX/VMS ANSI-Labeled Volume _____	B-3
B-2	Single File/Single Volume Configuration _____	B-6
B-3	Single File/Multivolume Configuration _____	B-7
B-4	Multifile/Single Volume Configuration _____	B-8
B-5	Multifile/Multivolume Configuration _____	B-9
B-6	Blocked Fixed-Length Records _____	B-16
B-7	Variable-Length Records _____	B-17

---

### TABLES

A-1	VAX/VMS Reserved Files _____	A-1
B-1	Labels and Components Supported by VAX/VMS _____	B-5



---

## Preface

The *Guide to VAX/VMS Disk and Magnetic Tape Operations* describes some of the routine tasks that are performed on private disk and magnetic tape media.

---

### Intended Audience

This guide is intended for all users, from the VAX operator to the VAX/VMS system manager. The only prerequisite is that *Introduction to VAX/VMS* be read first.

The *Guide to VAX/VMS Disk and Magnetic Tape Operations* is not a guide to system management. Although the system manager may use it as an aid in certain types of operations, this manual is designed primarily for the private user of disk and magnetic tape.

If you are a novice user, you can use this guide as a stepping stone to a basic understanding of disk and magnetic tape operations. Experienced users unfamiliar with DIGITAL software can use it to gain familiarity with DIGITAL terms and techniques. Experienced DIGITAL users should find the guide useful because it describes in detail most of the disk and magnetic tape operations routinely performed on the VAX/VMS operating system.

---

### Structure of This Document

Information in this guide is organized in four major sections:

- PART I describes basic disk and magnetic tape concepts. It also describes the protection schemes that apply to disk and magnetic tape media.
- PART II describes tasks that are frequently performed on private disk and magnetic tape media. The chapters in Part II include examples of command procedures designed to simplify the use of routine disk and magnetic tape operations.
- APPENDIX A describes the Files-11 On-Disk Structure for VAX/VMS disk files and volumes.
- APPENDIX B describes formats and components for VAX/VMS magnetic tape volumes and files.

## Associated Documents

The tasks described in this guide employ various VAX/VMS utilities and DCL commands. Detailed descriptions of the utilities and commands are provided in the VAX/VMS Utilities volumes and the *VAX/VMS DCL Dictionary*.

The *Guide to VAX/VMS Software Installation* provides instructions for procedures used in system installation.

The *VAX/VMS System Manager's Reference Manual* provides task-oriented instructions for maintaining public files and volumes; these instructions include steps for bootstrapping and running standalone BACKUP.

The *Guide to VAXclusters* provides information for maintaining public files and volumes in a VAXcluster environment.

The *Guide to VAX/VMS System Security* describes security features available through the VAX/VMS operating system; see this guide for more information on data protection.

The *VAX/VMS Volume Shadowing Manual* (not part of the VAX/VMS document set) describes how to mount, dismount, and maintain volumes using the volume shadowing option.

## Conventions Used in This Document

Convention	Meaning
<code>RET</code>	A symbol with a one- to six-character abbreviation indicates that you press a key on the terminal, for example, <code>RET</code> .
<code>CTRL/x</code>	The phrase CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key, for example, CTRL/C, CTRL/Y, CTRL/O.
<code>\$ SHOW TIME</code> <code>05-JUN-1986 11:55:22</code>	Command examples show all output lines or prompting characters that the system prints or displays in black letters. All user-entered commands are shown in red letters.

## Preface

Convention	Meaning
\$ TYPE MYFILE.DAT . .	Vertical series of periods, or ellipsis, mean either that not all the data that the system would display in response to the particular command is shown or that not all the data a user would enter is shown.
file-spec,...	Horizontal ellipsis indicates that additional parameters, values, or information can be entered.
[logical-name]	Square brackets indicate that the enclosed item is optional. (Square brackets are not, however, optional in the syntax of a directory name in a file specification or in the syntax of a substring specification in an assignment statement.)
quotation marks apostrophes	The term quotation marks is used to refer to double quotation marks ("). The term apostrophe (') is used to refer to a single quotation mark.



---

## **PART I DISK AND MAGNETIC TAPE CONCEPTS**



# 1

## Introduction to Disk and Magnetic Tape Media

This chapter provides an introduction to storage media and describes basic disk and magnetic tape concepts. It also mentions the importance of identifying routine disk and magnetic tape operations, and the advantage of using command procedures to perform them.

### 1.1 The History of File Storage Media

As technology has progressed, so has the volume of information that must be saved. Business and industrial concerns, for instance, have compiled many types of data files about a wide range of subjects. For a long time, all these data files were stored efficiently on paper in desk drawers and filing cabinets. But, as these paper files grew, it often took longer to locate the needed data than to create it in the first place. Storing all of a company's data files on paper became impractical.

As the need for saving data increased, the need also arose for a better medium to store and retrieve data quickly, reliably, and economically. The computer and computerized filing systems provide such storage media.

At first, computerized files consisted of collections of punched cards. Cards provide a convenient means of grouping related pieces of information. This information might represent, for example, a business event, such as a purchase or sale of office furniture. Or, in the engineering environment, the information could represent variable equations and data constants related to stress analysis. This information, grouped on a single card, represents a record of that event. Records of similar events, grouped together, constitute a file.

As a storage medium, cards have certain advantages. They are easy to add, delete, or rearrange. However, cards become worn, require physical handling, and are bulky. Cards are also relatively slow to process because they allow only sequential file access. Sequential access means that the search for a record starts at the beginning of the file and proceeds in order through each record. At times, when the needed record (or group of records) is near the end of the file, the search wastes computer processing time.

The introduction of magnetic tape as a storage medium eliminated some of the disadvantages of cards. Magnetic tape provides both a storage medium and a means for input/output (I/O), and its uses have grown along with the needs of the user. Magnetic tape offers virtually unlimited storage. It

requires much less storage space than a card file with a comparable amount of data. However, magnetic tape is limited as a storage medium because, like a card file, it allows only sequential file access.

In contrast to magnetic tape, disk storage allows direct file access. Direct access means that the computer can locate the desired record without first searching the records that precede it in the file. The time needed to access the record is independent of the record's location in the file.

---

### 1.2 Basic Disk Concepts

VAX/VMS disk files reside on Files-11 On-Disk Structure volumes. The term *Files-11 On-Disk Structure* refers to the logical structure given to the disk: a hierarchical organization of files, their data, and the directories needed to gain access to them. The VAX/VMS file system implements the disk structure and provides access control to the files located on the disk. This section describes the Files-11 On-Disk Structure levels and defines the terminology related to it. (The term *Files-11* used alone always refers to *Files-11 On-Disk Structure levels*.)

The smallest addressable unit of information on a disk is a block. Files-11 On-Disk Structures define a block to consist of 512 8-bit bytes. Blocks can be treated as units for transfer between a Files-11 disk volume and memory.

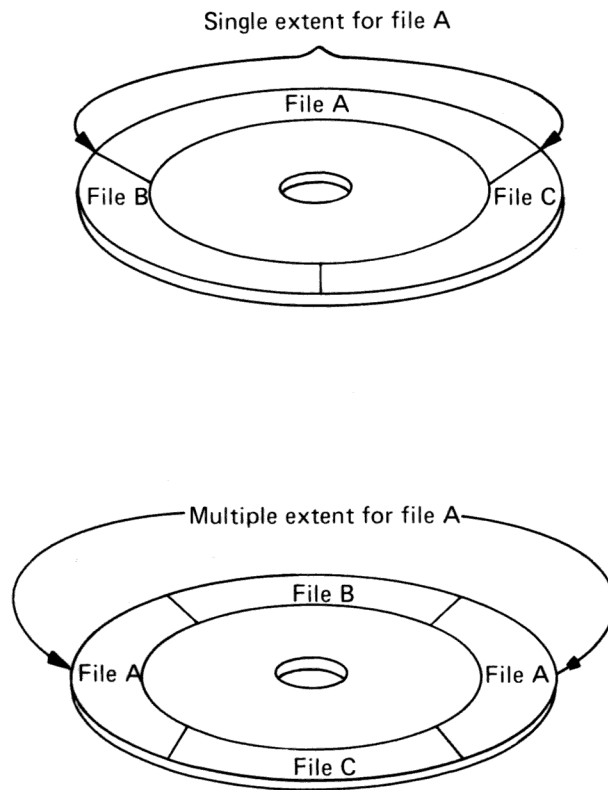
Blocks are logically grouped into a cluster, which is the basic unit by which disk space is allocated. The system manager or operator determines the number of blocks in a cluster when a given disk, known as a volume, is first prepared for use (initialized). Cluster sizes from 1 to 65,535 blocks are allowed, but the smaller cluster sizes in the range are more practical. In general, a disk with a relatively small number of blocks is given a smaller cluster size, while larger disks are given larger cluster sizes to minimize the overhead for disk space allocation.

Contiguous clusters allocated to a particular file have been given the name *extent*. An extent can contain all or part of a file. If enough contiguous area is available on the disk, the entire file is allocated as a single extent. Sometimes, however, not enough contiguous area is available to contain the entire file, or, when you create a file initially, you may not wish to reserve the entire required amount of space. When the file is eventually extended, it is unlikely that the adjacent clusters will still be unallocated. If the adjacent clusters are already allocated to another file, the extension will not occur contiguously. Whether the clusters are contiguous or not, the file is divided into two or more parts, and each part is an extent. Thus, a file can consist of multiple extents located in separate areas on the disk, as shown in Figure 1-1.

---

**Figure 1-1 File Extents**

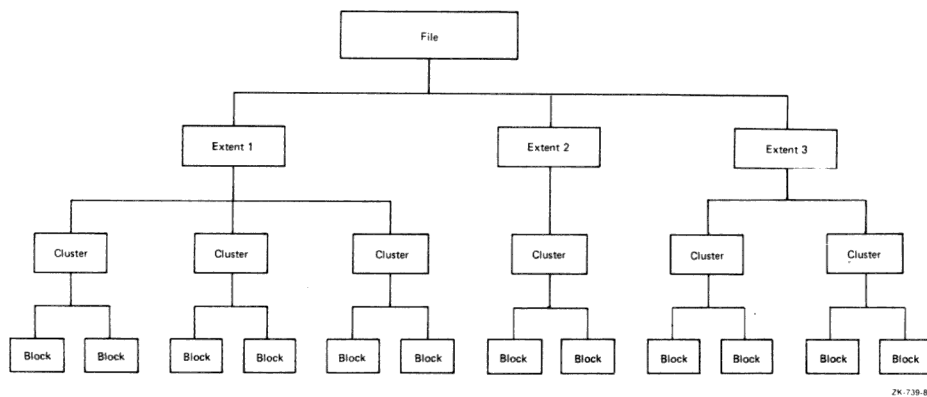
---



ZK-738-82

---

Figure 1-2 shows the hierarchy of blocks, clusters, extents, and files in the Files-11 On-Disk Structure.

**Figure 1-2 Files-11 On-Disk Structure Hierarchy**

You can exercise as little or as much control as you want over the allocation of space on a Files-11 disk file. For example, you can specify the number of blocks to be allocated, and even give the exact location for the blocks on the volume. On the other hand, you can allow VAX Record Management Services (RMS) to handle all disk space allocation details automatically.

If you wish, you can specify the size of the initial space allocation and the size to be used by RMS each time the file is extended. If you find that a file needs less space than is allocated to it, you can specify that the unused clusters are to be deallocated from the file. These clusters will then be available for allocation to other files on the volume.

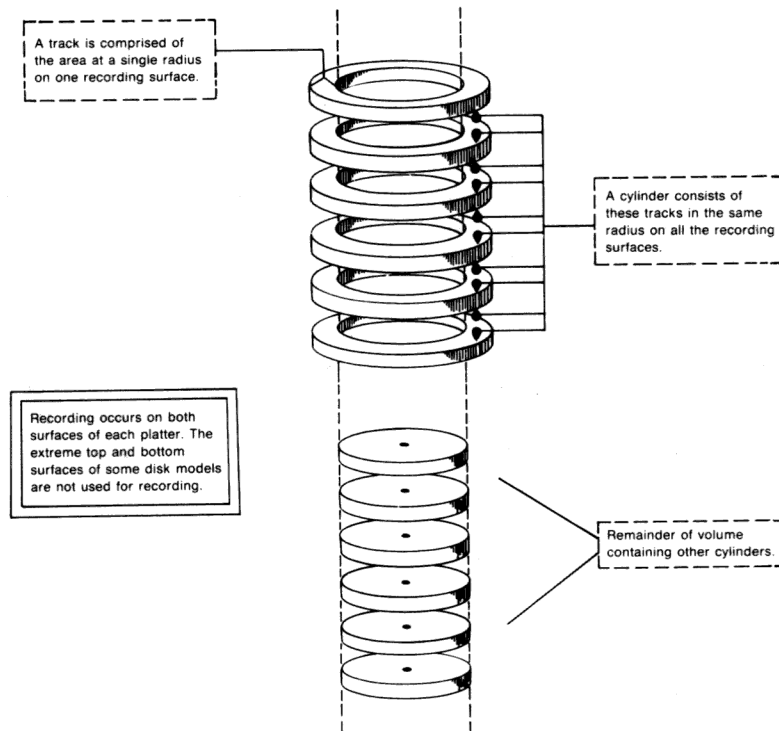
When a large amount of file storage space is needed, you can combine several Files-11 volumes into what is called a volume set. A volume set, although composed of several physical volumes, has the appearance of one large volume. The different extents of a file can be located on different volumes in the volume set. In general, you need not specify a particular volume in the set to locate a file or create a new one, although it is possible (and sometimes desirable for performance reasons) to specify a particular volume to be used for a certain allocation request.

The smallest unit discernible to the Files-11 structure is the sector; for most Files-11 disks, a sector is equivalent to a block, which is 512 bytes. Other basic terms related to disk structure are track and cylinder. A track is the collection of sectors (or blocks on Files-11 structures) at a single radius on one recording surface of a disk. It is accessible to a given read/write head position on the disk device. A cylinder consists of all tracks at the same radius on all recording surfaces of a disk.

Because access to any of the blocks in a given cylinder does not require any movement of the disk's read/write heads, it is generally advantageous to keep related data blocks in the same cylinder. For this reason, when choosing a cluster size for a large-capacity disk, a system manager will often select a cluster size that divides evenly into the cylinder size.

Figure 1-3 is a graphic representation of tracks and cylinders.

**Figure 1-3 Tracks and Cylinders**



A record is a collection of logically related data that a program treats as a processing unit. A record can be the same size as a block, smaller than a block, or even larger than a block. Therefore, a block can contain one record, multiple records, or only part of a record. The remainder of this section contains a brief explanation of some basic elements of the Files-11 structure.

A Files-11 structure resides on a volume, which is a physical medium, such as a disk pack. A Files-11 volume is an ordered set of 512-byte blocks. The blocks are numbered consecutively from 0 to  $n$ ; the value of  $n$  depends on the size of the disk.

Each Files-11 volume has an index file, which is created when the volume is initialized. (You cannot use a disk as a Files-11 disk until it has been initialized with the INITIALIZE command.) The index file contains the following information:

- Bootstrap block
- Home block
- File headers

The **bootstrap block** contains the bootstrap program and is physically the first block on the volume. All Files-11 volumes have an area for this bootstrap block even if the operating system does not require a bootstrap block. If the volume is not the operating system volume, the bootstrap block area contains a program other than the bootstrap program. This program displays a message on the system console indicating that the volume is not the operating system volume, but a volume containing only user files.

The **home block** is normally the next block after the bootstrap block; it identifies the disk as a Files-11 volume. If for some reason the home block cannot be read (physically unusable), an alternative block will be selected for use as the home block. This block provides specific information about the volume and default values for files on the volume. Among the items in the home block are the following:

- The volume name
- Information to locate the remainder of the index file
- The maximum number of files that can be present on the volume at any one time
- The user identification code (UIC) of the owner of the volume
- Volume protection information (specifies which users can read and/or write the entire volume)

Files-11 volumes contain several copies of the home block to ensure against accidental destruction of this information and the consequent loss of access to other files on the volume.

The bulk of the index file consists of **file headers**; each file header describes one file on the volume. File headers contain information such as the owner UIC, protection, creation date, and creation time. Most important, the file header contains a list of extents that make up the file, describing where the file is physically located on the volume. If a file has a large number of extents, multiple file headers are used to describe them. A file identifier number is associated with each file header.

When you create a file, you normally specify a file name to RMS, which assigns this name to the file on a Files-11 volume. RMS places the file name and file identifier associated with the newly created file in a directory, which contains an entry defining the location for each file. When you access the file, you supply the file name, which supplies a path to the file identifier through the directory entry. The file identifier, in turn, points to the location of the file header, which contains a listing of the extent or extents that locate the actual data.

---

### 1.3 Basic Magnetic Tape Concepts

The file storage system for magnetic tapes on VAX/VMS is based on the standard magnetic tape structure as defined by the American National Standard X3.27 - 1978. This standard will be referred to as the ANSI standard throughout this guide.

Magnetic tape data is organized sequentially; the data records and files are organized in the order in which they are written.

On VAX/VMS, characters of data on magnetic tape are measured in bytes per inch (bpi). This measurement is called density. (The ANSI standard uses characters per inch (cpi), which is equivalent to the bpi convention used by VAX/VMS.) A 1600 bpi tape can accommodate 1600 characters of data, using an inch of recording space.

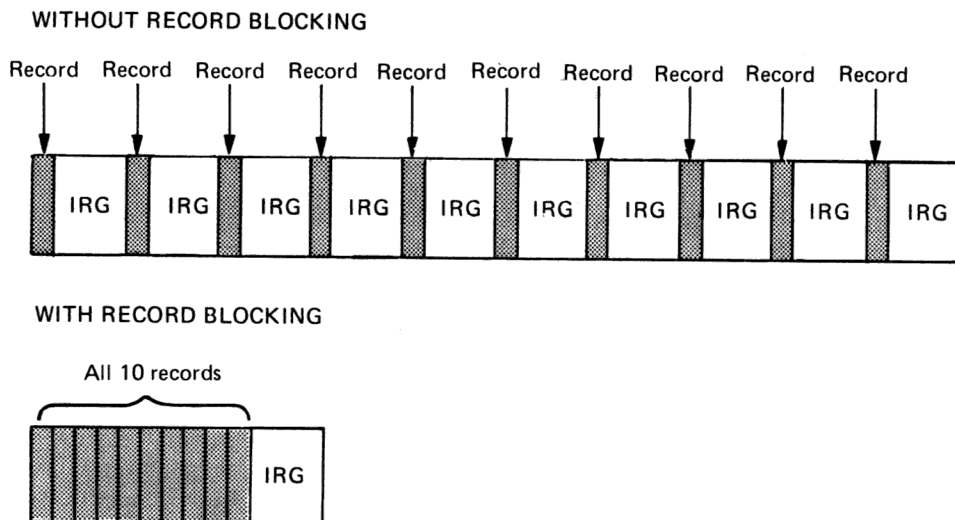
Even though a magnetic tape may have a density of 1600 bpi, there are not always 1600 characters on every inch of tape because of the interrecord gap (IRG). The IRG is an interval of blank space between data records; it is created automatically when records are written to the magnetic tape. The IRG is a breakpoint on the magnetic tape, which allows the magnetic tape unit to decelerate, and stop if necessary, after a record operation and then accelerate to working speed before the next record operation.

Each IRG is approximately 0.6 inch in length. Writing an 80-character record at 1600 bpi requires 0.05 inch of space. Therefore, the IRG requires 12 times more space than the data, wasting valuable storage space. RMS can reduce the size of this wasted space by using record blocking. This technique groups individual records into a block and places the IRG after the block rather than after each record.

Note that a block on disk is different from a block on magnetic tape. On disk, a block is fixed at a size of 512 bytes; on magnetic tape, the size of a block is determined by the user. However, record blocking requires more buffer space to be allocated for your program, which increases RMS memory requirements. The greater the number of records in a block, the greater the buffer size requirements. You must determine the point at which the benefits of record blocking cease. This determination should be based on the configuration of your computer system.

The following example shows how space can be saved by record blocking. Assume that a 1600-bpi magnetic tape contains 10 records that are not grouped into a block. Each record is 160 characters long (0.1 inch at 1600 bpi) with a 0.6 inch IRG after each record. This uses 7 inches of tape. However, placing the same 10 records into one block uses only 1.6 inches of tape (1 inch for the data records and 0.6-inch for the IRG). Figure 1-4 illustrates this example of record blocking.

**Figure 1-4 Interrecord Gaps**



ZK-741-82

Record blocking also increases the efficiency of the flow of data into the computer. For example, 10 unblocked records require 10 separate physical transfers, while 10 records placed into a single block require only 1 physical transfer. Moreover, a shorter length of magnetic tape is traversed for the same amount of data; thus the operation is completed in less time.

Data on magnetic tape is also organized into files. When you create a file on magnetic tape, the magnetic tape file system writes a set of header labels on the tape immediately preceding the data blocks. These labels contain information such as the user-supplied file name, creation date, and expiration date. Additional labels, called trailer labels, are also written following the file. To access a file on magnetic tape by the file name, the file system searches the tape for the header label set that contains the specified file name.

When the data blocks of a file or related files do not physically fit on one volume (a reel of magnetic tape), the file is continued on another volume. This is a multivolume file. The volumes in a multivolume file constitute a volume set. When access is made to a file on a volume set, all volumes in the set are searched.

---

### 1.4 How to Identify Routine Operations

Many of the operations that you perform on disk and magnetic tape media are routine in nature. You will find it worthwhile to take the time to identify those tasks that you routinely perform at your particular site. Once you have isolated those tasks, you can design command procedures to assist you in performing them.

If you are a system manager or an operator, for example, you must frequently perform data integrity tasks such as backing up media. You could enter all of the commands, parameters, and qualifiers required to back up your media each time that you perform the backup operation. Or, you could write a single command procedure (containing that set of commands, qualifiers, and parameters) that, when executed, would also perform the backup operation.

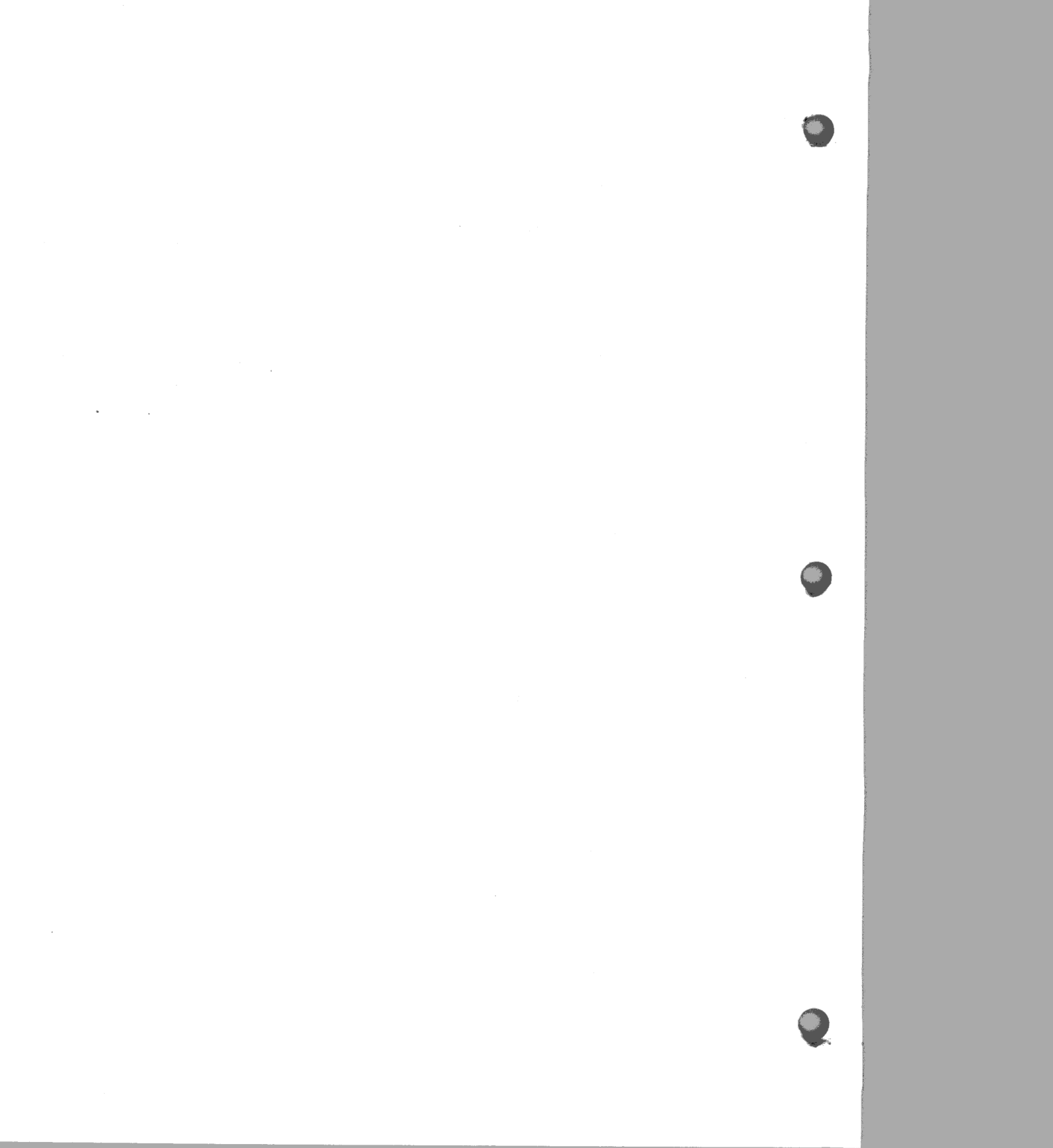
---

### 1.5 Using Command Procedures to Perform Routine Operations

In order to familiarize yourself with the syntax used to design and execute command procedures, see the *Guide to Using DCL and Command Procedures on VAX/VMS*.

Each of the chapters in Part II of this guide describes a routine type of disk and magnetic tape operation. With the exception of Chapter 7, each of these chapters contains examples of command procedures that are designed to ease the performance of routine disk and magnetic tape operations.

Chapter 7 specifically describes operator-oriented disk and magnetic tape functions.



# 2

## Disk and Magnetic Tape Protection

This chapter describes the types of data protection that the VAX/VMS operating system provides for disks and magnetic tapes. In particular, the chapter focuses on disk and magnetic tape protection at the volume level and at the file level.

### 2.1 Data Protection

The VAX/VMS operating system protects data on disk and magnetic tape volumes to ensure against accidental or unauthorized access. A volume is the entity that exists when a medium is mounted on a device. For example, disk packs and reels of magnetic tape are called volumes when they are mounted on disk and magnetic tape drives.

VAX/VMS systems support the protection of data on disk and magnetic tape media at the volume and file levels. At the volume level, the system provides protection for both disks and magnetic tapes. At the file level, the system provides protection for individual disk files, directory files that reside on disk volumes, and magnetic tape files. However, distinct file-level protection for magnetic tape files is limited to one special case, which is described in Section 2.3.3.

In addition to protecting the data on mounted volumes, the system also provides device-level protection. Note, however, that this chapter is primarily concerned with how to protect data residing on disk and magnetic tape media. It does not describe how to set device protection. For information on setting device protection characteristics, see the descriptions of the DCL commands INITIALIZE, MOUNT, and SET VOLUME in the *VAX/VMS DCL Dictionary*.

On VAX/VMS, data residing on disk and magnetic tape volumes can be protected by one or more of the following:

- user identification codes (UICs)
- access control lists (ACLs)
- ANSI-standard accessibility

### 2.1.1 UIC-Based Protection

UIC-based protection is supported for disk and magnetic tape volumes and individual files on disk volumes, including directories. UIC-based protection is determined by an owner's user identification code (UIC) and a protection code. The owner UIC is normally the UIC of the user who created the file or volume. The protection code indicates who is allowed access and for what purposes.

UIC formats can contain alphanumeric characters or they can consist entirely of octal numbers. For example, [VMS,USER] and [360,030] are both legitimate UIC formats.

When a user attempts to access a file or volume, the user's UIC is compared to the owner UIC of the file or volume. Depending on the relationship of the UICs, the user falls into one or more of the following categories:

- **SYSTEM** — all users who have system privilege (SYSPRV) or low group numbers, usually from 1 through 10 (octal). However, the exact range of system group numbers is determined by the system manager when the system is generated and may range as high as 37,776 (octal). These group numbers are generally for system managers, system programmers, and operators. In addition, the SYSTEM protection field is used for those users with the user privilege GRPPRV, whose UIC group matches the group of the file or volume owner UIC.
- **OWNER** — the user with the same UIC as the person who created and therefore owns the volume or file
- **GROUP** — all users, including the owner, who have the same group number in their UICs as the owner of the file
- **WORLD** — all users including those in the first three categories

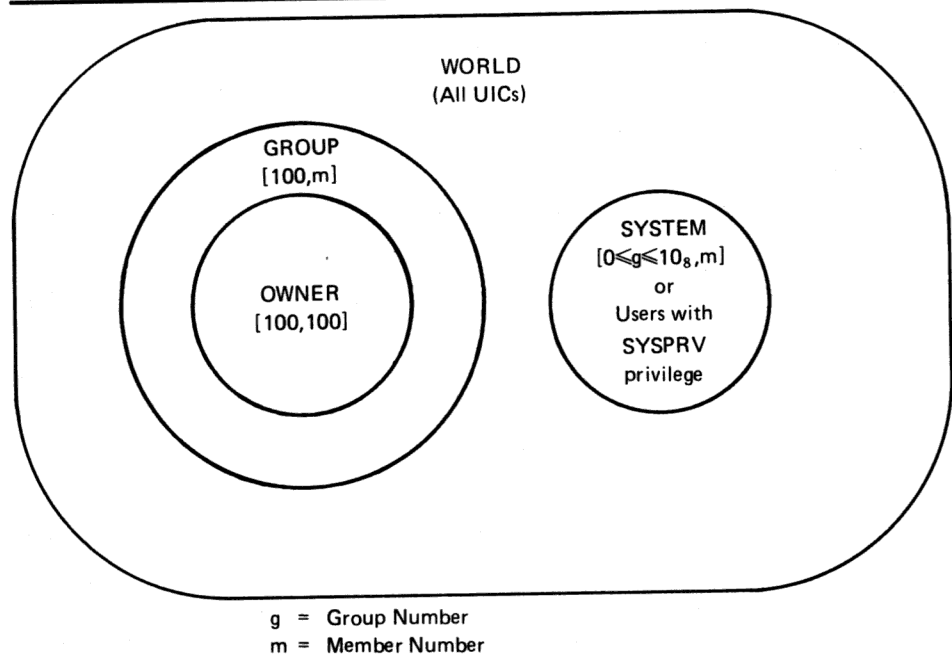
#### Note

**SYSTEM and OWNER categories always have access to magnetic tape volumes, except as noted in Section 2.1.3.**

Figure 2-1 illustrates the relationships of these categories to each other.

Each of the categories of user described above is allowed or denied several types of access, including the following:

- READ
- WRITE
- EXECUTE

**Figure 2-1 Illustrating User Categories with a UIC of [100,100]**

NOTE: THE SYSTEM MANAGER CAN EXTEND THE SYSTEM GROUP NUMBER LIMIT TO  $3776_8$

ZK-778-82

- DELETE
- CONTROL

The meaning of each access type depends on the object to which it is applied (disk or magnetic tape volume, file, or directory). Section 2.2 describes how these types of access apply to disk and magnetic tape volumes; Section 2.3 describes how the access types apply to files and directories.

Note that CONTROL access does not appear in the protection mask. It is never granted to the GROUP and WORLD categories of users, but is always granted to SYSTEM and OWNER.

You can bypass all UIC-based protection checks if you have the BYPASS user privilege. In addition, you can bypass all protection checks and gain READ and CONTROL access to a disk file if you have the READALL user privilege. For magnetic tapes, you can bypass UIC-based protection checks if you have VOLPRO (volume protection override) privilege.

For information on how to set UIC-based protection, see the discussion of protection codes in the *VAX/VMS DCL Concepts Manual*.

### 2.1.2 ACL-Based Protection

An access control list (ACL) is primarily a list of entries that grant or deny access to a particular system resource, such as a disk file or a directory. ACLs offer you the chance to "fine tune" the action taken when access to a file is sought. You can opt to provide an ACL on any disk file to permit as much or as little access as is desirable in each case. In providing a more detailed definition of who is allowed a particular kind of access, ACLs can enhance the security of disk files. Note, however, that "fine tuning" is done at the expense of performance; the larger the ACLs are, the more time they take to process.

ACL-based protection is supported for files and directories on disk volumes only. ACL protection is not supported for files on magnetic tape volumes.

If a file does not have an associated ACL, the system uses UIC-based protection to determine access as described in Section 2.1.1.

If a file has an associated ACL, the system uses the ACL to determine access as follows:

- If the ACL includes an identifier entry for your UIC, the ACL specification takes precedence over your UIC-based protection. However, SYSTEM and OWNER may still have access (for detailed information on ACL protection, see the *VAX/VMS DCL Concepts Manual*).
- If the ACL does not include an identifier entry for your UIC (that is, does not explicitly allow or refuse access), then the system uses UIC-based protection to determine access.

If you have control access to a resource, you can define the access control list using the following DCL commands:

- EDIT/ACL
- SET ACL

For more information on how to invoke, modify, and display ACLs, see the *VAX/VMS Access Control List Editor Reference Manual* or the *Guide to VAX/VMS System Security*.

### 2.1.3 VAX/VMS ANSI-Labeled Magnetic Tape Accessibility Protection

The VAX/VMS magnetic tape file system supports accessibility protection based on the ANSI standard. This protection scheme allows an installation to use a routine that is designed to interpret the contents of the volume and header label accessibility field (see Appendix B).

When the installation routine is called to interpret a VOL1 label supplied as input, the routine must return one of the following values to the magnetic tape file system:

- User has no access to the volume/file
- User has complete access to the volume/file
- Use the output of any other VAX/VMS protection mechanism specified (for file-level access, the protection defaults to whatever was specified for the volume)

If the installation routine determines that you have no access to a volume or file, then you will be denied access regardless of the volume protection set by VAX/VMS. That is, no amount of privilege granted to you by VAX/VMS can override that decision. On the other hand, if the installation determines that you have complete access to the file or volume, then you will be granted complete access even if a VAX/VMS protection scheme denies you access to that file or volume.

When the installation routine is called to return the VOL1 label or HDR1 label as output, the routine must return the character to write into the volume or file label accessibility field.

If you do not design your own installation routine, VAX/VMS will provide a default routine for you. The routine provided by VAX/VMS will work in the following way. The default installation routine first checks the ANSI standard version of the label. For magnetic tapes with a version number of 3 or less, the routine outputs either an ASCII character space (full access without checking VAX/VMS protection) or the character specified by the user. On input of these magnetic tapes, the routine checks for a blank (full access without checking VAX/VMS protection) and, if the field is not blank, returns the value that causes the file system to check for explicit override of accessibility checking.

For magnetic tapes with a version number greater than 3, the routine outputs either the character specified by the user or an ASCII 1 if no character was specified. On input of these magnetic tapes, the routine checks for an ASCII character space. If the field has an ASCII space, the user is given full access and VAX/VMS protection is not checked. If the field contains an ASCII 1 and the VAX/VMS protection has been specified, the VAX/VMS protection is checked.

If the field does not contain an ASCII space or an ASCII 1, the routine returns the value to the magnetic tape file system that forces the user to override the accessibility checking, and allows the magnetic tape file system to check VAX/VMS protection.

For more information on how ANSI-standard accessibility protection applies to files on magnetic tape volumes, see Section 2.3.3.

---

## 2.2 Volume-Level Protection

Disks and magnetic tapes can both be protected at the volume level. When you prepare a disk or magnetic tape volume for private use, you can define the protection you want to have applied.

Volume protection for a disk or magnetic tape volume is usually set when the volume is mounted or initialized. If you do not explicitly specify the protection for a particular disk volume, the system will provide a default protection for you.

To change the protection that has been set on a disk volume, use the DCL command SET VOLUME as described in the *VAX/VMS DCL Dictionary*. For more information on setting the volume protection when you mount a volume, see the descriptions of the DCL command INITIALIZE in the *VAX/VMS DCL Dictionary*, and see the *VAX/VMS Mount Utility Reference Manual*.

---

### 2.2.1 Disk Volume Protection

Disk volume protection supports the following access types:

- READ
- WRITE
- EXECUTE
- DELETE

If you have READ access to a disk volume, you have the right to examine, print, execute, or copy files on that volume.

If you have WRITE access to a disk volume, you have the right to modify existing files on that volume. Unlike that of magnetic tape volumes, WRITE access to a disk volume does not automatically imply READ access. That is, it is possible to grant a user WRITE access to a disk volume, without granting that user READ access to the volume. This is not very useful, however, since WRITE access to a file implies READ access.

When applied at the volume (as opposed to the file) level, EXECUTE access means that you have the right to create files or modify existing files. When applied at the file level, EXECUTE access gives you the right to invoke files that are executable (such as executable images and command procedures).

If you have DELETE access to a disk volume, you have the right to delete a file or files on the volume.

---

### 2.2.2 Magnetic Tape Volume Protection

Only two types of access apply to magnetic tape volume protection, unlike disk volume protection. These two access types are:

- READ
- WRITE

If you have READ access to a magnetic tape volume, you have the right to examine, print, or copy a file or files from the volume. If you have WRITE access to a magnetic tape volume, you have the right to append or write files to the volume.

For magnetic tape volumes, WRITE access implies READ access. Thus, granting a category of users WRITE access to a magnetic tape volume automatically permits them to have READ access to the volume also.

There are two ways to protect a magnetic tape volume on a VAX/VMS system. You can protect a magnetic tape volume through the UIC-based protection scheme supported by VAX/VMS. This scheme is checked on VAX/VMS systems only and will be ignored in any interchange with non-VAX/VMS systems. You can also protect a magnetic tape volume by using the guidelines of the ANSI standards. This protection scheme supports the protection of magnetic tape volumes in environments where interchange exists between VAX/VMS and non-VAX/VMS operating systems.

---

#### 2.2.2.1 Protection for VAX/VMS Magnetic Tapes

The VAX/VMS magnetic tape file system provides two levels of protection. One level is defined by the ANSI standard and is encoded in the ACCESSIBILITY field of the first volume label written on the magnetic tape. The second level is UIC-based and is defined by the VAX/VMS magnetic tape file system. This protection is encoded in the second volume label written on the magnetic tape.

---

### 2.2.2.2 Protection for Interchange Environments

Magnetic tape volume protection is also supported for interchange between VAX/VMS and non-VAX/VMS operating systems. Protection is supported for interchange between VAX/VMS and DIGITAL operating systems other than VAX/VMS (such as RSX-11M and TOPS-20) as well as between VAX/VMS and non-DIGITAL operating systems.

For magnetic tapes processed on any operating system that supports a version of the ANSI standard later than Version 3, the accessibility information in the first volume label is processed exactly as described above. Magnetic tapes processed on operating systems other than VAX/VMS Version 4.0 or later have their protection characteristics encoded in the first volume label of the magnetic tape volume.

In order to process a magnetic tape created on a DIGITAL operating system other than VAX/VMS, a user must have VOLPRO privilege and must explicitly override the check on the protection. If the magnetic tape had been created with a specified accessibility, then a user wishing to access that tape must have the appropriate privilege and must explicitly override the check on accessibility. If the magnetic tape volume had not been created with such a protection scheme, then a user wishing to access that magnetic tape would be granted READ and WRITE access to that magnetic tape volume.

The VAX/VMS magnetic tape file system allows you to specify values for the fields in which other DIGITAL operating systems currently write their protection information. Except under conditions described in the two previous sections, VAX/VMS will not process this field. This field can thus be used to store the protection values for another operating system without affecting VAX/VMS protection characteristics on that particular volume.

---

## 2.3 File-Level Protection

In addition to volume-level protection, the VAX/VMS operating system supports protection at the file level for all files residing on disk volumes, including directories. In certain cases, file-level protection is also supported for magnetic tape files. For files residing on magnetic tape volumes, however, only one special case (described in Section 2.3.3) exists in which distinct file-level protection applies.

File-level protection is described as it applies to each of the following:

- Disk file protection
- Directory file protection
- Magnetic tape file protection

---

### 2.3.1 Disk File Protection

Each file on a disk has its own protection code, which is distinct from the protection that applies to the disk volume itself.

For files residing on disk volumes, the following access types are supported:

- READ
- WRITE
- EXECUTE
- DELETE
- CONTROL

If you have READ access to a file or group of files on a disk volume, you have the right to examine, print, or copy that file or group of files. READ access automatically includes EXECUTE access to a specified file or group of files on disk.

If you have WRITE access to a file or group of files on a disk volume, you have the right to modify the file or group of files. As with disk volume protection, it is possible to be granted WRITE access without having READ access. This is not very useful however, since opening a file for a write operation implies READ access.

If you have EXECUTE access to a disk file or group of disk files, you have the right to execute that file or group of files containing executable program images or DCL command procedures.

If you have DELETE access to a particular disk file or group of disk files, you have the right to delete the file or group of files.

If a user has CONTROL access to a particular disk file or group of disk files, that user has the right to change the characteristics of the file or group of files.

You can specify a protection code when you create or copy a file by using the /PROTECTION qualifier, in the following example:

```
* COPY/PROTECTION=(SYSTEM:RW,OWNER:RWED,GROUP:RW,WORLD)
```

You can also change the protection for an existing file by using the SET PROTECTION command. For example, you can type the following:

```
* SET PROTECTION=(SYSTEM:RWE,OWNER:RWED,GROUP:RE,WORLD) ABC.DAT
```

This command gives VAX/VMS the following instructions regarding the file in question (ABC.DAT): SYSTEM has READ, WRITE, and EXECUTE privileges; OWNER has READ, WRITE, EXECUTE, and DELETE privileges; GROUP has READ and EXECUTE privileges only; and WORLD has no access.

For SYSTEM and OWNER, CONTROL access is implied and unchangeable and not so for GROUP and WORLD.

If you do not define a protection code for a file when you create the file, the system applies a default protection. If a version of the file already exists, protection is taken from the previous version of the file. For a new file, protection is determined in one of the two following ways:

- If the directory where the file is to be cataloged has an associated access control entry that specifies the DEFAULT\_PROTECTION entry, then the specified protection is used.
- If the directory does not have the DEFAULT\_PROTECTION entry, then the default process protection is used. (The default process protection is established explicitly with the SET PROTECTION/DEFAULT command, or by default when you log in.

You can determine the current default protection by issuing the SHOW PROTECTION command:

```
$ SHOW PROTECTION
SYSTEM=RWED, OWNER=RWED, GROUP=RE, WORLD=NO ACCESS
```

This response is the system default protection. It indicates that SYSTEM and OWNER have all types of access, members of the owner's group have READ and EXECUTE access, and all other users (WORLD) have no access.

To determine the current protection associated with a specific file or files, you can use the /PROTECTION qualifier on the DCL command DIRECTORY. For example:

```
$ DIRECTORY/PROTECTION MYFILE.REC
Directory DBA1:[CRAMER]
MYFILE.REC;5      (RWED,RWED,RW,R)
Total of 1 file.
```

You can change the default protection applied to files that you create during an interactive session by using the SET PROTECTION/DEFAULT command. The SET PROTECTION/DEFAULT command indicates that the protection code you specify is to be applied to all files that you subsequently create during the terminal session or batch job (providing that the files are not subject to other sources of protection).

Note that to completely protect a file you must protect both the file itself and the directory in which the file is listed. If you have files that must be protected against unauthorized access, be sure to specify the proper protection both for the directories in which the files are listed and for the files themselves.

---

### 2.3.2 Directory File Protection

Each directory file has a protection associated with it. The directory protection can override the protection of individual files in the directory. For example, if a directory denies WORLD access, WORLD users cannot look up even those files in the directory that permit WORLD access.

For directory file protection, the following access types are supported:

- READ
- WRITE
- EXECUTE
- DELETE
- CONTROL

Having READ access to a file in a directory means that you have the right to examine, print, or copy that file. If you have READ access to a directory file, you can display the contents of the directory file with the DIRECTORY command. For example, if you have access to the directory [JONES], you can issue the following:

```
* DIRECTORY [JONES]
```

This command generates a display of the files contained in the [JONES] directory.

If you have READ access, you can access any file listed in the directory, unless the protection on that file denies you access. If the protection applied to the whole directory denies you READ access, then you cannot access even those files in the directory that permit access to users in your group.

If you have WRITE access to a directory file, you have the right to modify or write to that directory file. However, you must have both READ and WRITE access to a directory in order to create files in that directory, to rename files in that directory, or to perform any file operation that involves changes to that directory file.

Note that EXECUTE access has a special meaning when applied to directories. EXECUTE-only access to a directory file allows you to use the DIRECTORY command to look up files that you can identify by name. It also means that you can access files in the directory that are not protected against users in your category, provided that you do not perform an operation that modifies the directory file. However, you cannot list all the entries in a directory by using wildcards.

For example, if you have EXECUTE-only access to the [JONES] directory, you issue the command:

```
# DIRECTORY [JONES]
```

The system responds with an error message and does not list the files in the [JONES] directory. However, if you know that the file DATAFILE.DAT resides in the [JONES] directory, you can issue the command:

```
# TYPE [JONES]DATAFILE.DAT
```

The system displays the file. Therefore, EXECUTE-only access provides some, but not all, of the operations that READ access provides.

If you have DELETE access to a directory file, you have the right to delete that directory file. You must remove all entries from a directory file before you can delete it. When you create a directory file with the CREATE /DIRECTORY command you do not, by default, get DELETE access. If you want to be able to delete a directory file, you must use the DCL command SET PROTECTION to explicitly assign DELETE access to the OWNER category.

If you have CONTROL access to a directory file, you have the right to change the characteristics of the directory file.

Remember that to ensure that your disk files are adequately protected, you must make certain that they are properly protected at both the directory and file levels. ACLs allow you to further control access to disk files. For more information on ACLs, see the *VAX/VMS Access Control List Editor Reference Manual*.

---

### 2.3.3 Magnetic Tape File Protection

In general, the protection that applies to a magnetic tape volume will also apply to all the files on that volume. For VAX/VMS ANSI-labeled magnetic tapes, however, distinct file-level protection is also supported. For VAX/VMS ANSI-labeled magnetic tapes, file-level protection can be determined by the contents of the accessibility field, which is located in the first header label of each file.

Each time that a VAX/VMS ANSI-labeled magnetic tape file is opened for processing, the installation routine examines the contents of the accessibility field of that file. Depending on the value returned by this routine, access to the file in question will either be granted, denied, or defaulted to the VAX/VMS protection scheme applied to the volume containing that file (see Section 2.1.3).



---

## **PART II DISK AND MAGNETIC TAPE OPERATIONS**



# 3

## Preparing Disk and Magnetic Tape Volumes for Private Use

---

This chapter describes how to prepare private disk and magnetic tape volumes for routine operations. The DCL commands **ALLOCATE**, **INITIALIZE**, and **MOUNT**, as well as many of the qualifiers that apply to each of these commands, are discussed. This chapter also contains examples of command procedures that are designed to set up disk and magnetic tape volumes for routine processing.

If you are interested in setting up public volumes, see the description in the *VAX/VMS System Manager's Reference Manual*.

---

### 3.1 Setting Up a Private Volume

Under some circumstances, it may be desirable for you to perform your work on a device that cannot be accessed by unauthorized users. By creating a private volume and mounting it on a device allocated exclusively to your process, you can perform your work without fear of interference from other users.

You can set up your private volume by performing the following steps:

- Use the **ALLOCATE** command to assign a disk or magnetic tape drive exclusively to your process.
- Use the **INITIALIZE** command to format the volume and write an identifying label on it.
- Use the **MOUNT** command to make a volume, and the files or data it contains, accessible to your process.

Each of these DCL commands is discussed more thoroughly in the sections that follow. See the *VAX/VMS DCL Dictionary* for complete descriptions of the DCL commands **ALLOCATE** and **INITIALIZE**; for a description of the Mount Utility (**MOUNT**), see the *VAX/VMS Mount Utility Reference Manual*.

---

## 3.2 Allocating a Drive to Your Process

Before you can begin processing files or data on a private volume, you must first allocate a drive to your process. You can use the DCL command **ALLOCATE** to logically assign a disk drive or a magnetic tape drive to your process.

Whether you are allocating a disk drive or a magnetic tape drive, the format for the **ALLOCATE** command is as follows:

```
ALLOCATE device-name[:] [logical-name]
```

### Command Parameters

#### **device-name[:]**

Specifies the drive on which the volume will be loaded. The device name can be a physical, generic, or logical name. A physical device name consists of a device code, alphabetic controller designation, and a unit number. A generic device name consists only of the device code. A logical name must equate to a physical or generic name. Use of the colon is optional but recommended by DIGITAL.

#### **[logical-name]**

Specifies an optional logical name to be associated with the specified disk or magnetic tape drive.

The **ALLOCATE** command allocates only one device to a process. Except for a list of generic device names, **ALLOCATE** does not accept lists of device names in the command string. Although you can specify a list of generic devices, the first available device will be the only one to be allocated.

The next two sections contain examples of how to use the **ALLOCATE** command to allocate disk drives and magnetic tape drives.

---

### 3.2.1 Allocating a Disk Drive

This section includes examples of how to use the **ALLOCATE** command to allocate disk drives. Although the examples focus on the allocation of RK06/RK07 drives, they are intended to apply to other disk devices as well.

---

## EXAMPLES

1     \$ ALLOCATE DM: DISK

In this example, the ALLOCATE command requests that the first available RK06/RK07 be assigned to your process. The logical name DISK is placed in your process logical name table and assigned the name of the allocated device. Other users are unable to access the device.

2     \$ ALLOCATE \_DMB2:  
DCL-I-ALLOC, \_MARS\$DMB2: allocated

In this example, the ALLOCATE command requests the allocation of a specific RK06/RK07 disk drive, that is, unit 2 on controller B. The response from the ALLOCATE command indicates that the device was successfully allocated.

If you wish to allocate a particular type of device, you can use the /GENERIC qualifier with the ALLOCATE command. For example, device DM could be an RK06 or RK07 disk. If you specifically want to allocate an RK07, you would use the /GENERIC qualifier in the following way:

      \$ ALLOCATE/GENERIC RK07 MYDISK

In this case, the system would allocate the first available RK07 device to your process. For further discussion of the /GENERIC qualifier as well as all of the qualifiers applicable to the ALLOCATE command, see the *VAX/VMS DCL Dictionary*.

---

### 3.2.2 Allocating a Magnetic Tape Drive

This section contains examples of how to use the ALLOCATE command to allocate magnetic tape drives. It also shows how you can use the /GENERIC qualifier with the ALLOCATE command to allocate a specific type of magnetic tape device.

---

## EXAMPLES

1     \$ ALLOCATE MTA1:  
%DCL-I-ALLOC, \_MARS\$MTA1: allocated

In this example, the ALLOCATE command specifies a physical device name MTA1:.

VAX/VMS then informs you that MTA1 has been allocated.

2 \$ ALLOCATE MF,MT,MS DRIVE  
%DCL-I-ALLOC, \_MARS\$MTA0: allocated

In this example, the ALLOCATE command specifies a list of generic device names. At a minimum, a generic device name consists of the device code; a controller designation is optional. Only one of the specified generic devices is allocated. Each element in the list must represent a unique generic device type.

VAX/VMS then informs you that drive MTA0: has been allocated. Although it is not indicated in the message, VAX/VMS also assigns the logical name DRIVE to the drive MTA0:.

3 \$ ALLOCATE DRIVE1: D1  
%DCL-I-ALLOC, \_MARS\$DBA3: allocated

In this example, the ALLOCATE command specifies a logical name DRIVE1 as the device name. (This example assumes that DRIVE1: has already been defined.)

VAX/VMS then informs you that DRIVE1: has been allocated. Although it is not indicated in the message, VAX/VMS also assigns the new logical name D1 to the drive DRIVE1:.

If you wish to allocate a specific type of magnetic tape device, you can use the /GENERIC qualifier. For example, if you want to allocate a TU78 device specifically, you would use the /GENERIC qualifier with the ALLOCATE command, as follows:

\$ ALLOCATE/GENERIC TU78 TAPE\_TU78

In this case, the system would allocate the first available TU78 device to your process. For a further discussion of the /GENERIC qualifier as well as all of the qualifiers applicable to the ALLOCATE command, see the *VAX/VMS DCL Dictionary*.

---

### 3.3 Initializing a Volume

Before you can write files or data to a disk or magnetic tape volume, the volume must be initialized. You can use the DCL command INITIALIZE to format and write a label to the volume.

The INITIALIZE command does the following:

- Invalidates all existing data on the volume, if any, and creates a new file structure
- Writes a label on the volume to identify it
- Defines the owner UIC and the protection for the volume

Whether you are initializing a disk or magnetic tape volume, the format for the INITIALIZE command is as follows:

```
INITIALIZE device-name[:] volume-label
```

### Command Parameters

#### **device-name[:]**

Specifies the name of the device on which the volume is to be physically mounted and then initialized. To prevent initializing another person's volume, you should allocate a device before you initialize the volume. Prior allocation is not required, however.

#### **volume-label**

Specifies the identification to be encoded on the volume. Characters are automatically changed to uppercase. You can specify up to 12 alphanumeric characters for a disk volume, or up to 6 alphanumeric characters for a magnetic tape volume. Nonalphanumeric characters other than "\$" and "\_" are invalid for disk volume-label specification.

There are some cases in which you might be prevented from accessing and initializing a particular volume. For example, if the volume that you wish to initialize previously contained data, the protection code may prevent you from accessing and initializing that particular volume. In the case of a magnetic tape volume, you may not be able to initialize the volume if the first file on the volume has not reached its expiration date, or if the volume or file accessibility is such that the installation routine provided by VAX/VMS does not allow you to access the volume.

If the volume is protected or if the expiration date on the first file has not been reached, and you are not the owner or a SYSTEM user, you must have VOLPRO user privilege to override volume protection. If you do not have VOLPRO privilege, you can ask the previous owner of the volume or another user (the system manager or operator, for example) who does have READ/WRITE access to initialize it for you. If the installation routine provided by VAX/VMS (or a user-designed installation routine) does not allow you access, then consult your system manager.

When you give the volume to another user for initialization, you should specify the following:

- The label you want to have written on the volume
- The protection code and owner UIC you want assigned to the volume

When you obtain a magnetic tape or disk volume, place identification on the outside of the volume so that it can be easily identified.

The next two sections describe how to initialize disk and magnetic tape volumes.

### 3.3.1 Initializing a Disk Volume

This section includes two examples of how to use the INITIALIZE command to initialize a disk volume.

By default, the INITIALIZE command builds a Files-11 structure on your new volume. The default format for disk volumes in the VAX/VMS operating system is called the Files-11 On-Disk Structure Level 2. The INITIALIZE command can also initialize disk volumes in the Files-11 On-Disk Structure Level 1 format.

You do not need special privileges to override logical protection on a blank disk volume (that is, a volume that has never been written to) or a disk volume that is owned by your current UIC or by UIC [0,0]. In all other cases, you must have the user privilege VOLPRO to initialize a disk volume.

The following examples include typical cases of initializing a disk.

#### EXAMPLES

1 \$ INITIALIZE DISK USER\_DISK

In this example, the volume is given the label USER\_DISK. DISK is the logical name of the device on which it is mounted.

2 \$ ALLOCATE DJA2: TEMP  
%DCL-I-ALLOC, \_MARS\$DJA2 allocated  
\$ INITIALIZE TEMP: BACKUP\_FILE

This example shows how to initialize an RA60 volume. First, the drive is allocated, to ensure that no one else can access it. Then, when the volume is physically loaded on the drive, the INITIALIZE command initializes it.

### 3.3.2 Initializing a Magnetic Tape Volume

This section describes how to use the INITIALIZE command to initialize a magnetic tape volume.

The default format for magnetic tape volumes in the VAX/VMS operating system is based on Level 3 of the ANSI standard for magnetic tape labels and file structure for informational interchange (ANSI X3.27-1978).

You can use the DCL command INITIALIZE to encode VAX/VMS ANSI-labeled format on a magnetic tape volume. INITIALIZE writes labels to an empty file on the magnetic tape volume in the following order:

- 1 A volume label
- 2 File header labels with the file sequence number set to 0

- 3 Two tape marks framing an empty file (BOT and EOT)
- 4 Corresponding end-of-file labels (EOF)
- 5 A double tape mark, specifying logical end-of-volume

The following example describes how to initialize a magnetic tape volume.

**\$ INITIALIZE TAPE USER**

In this example, the magnetic tape volume is given the label USER. TAPE is the logical name of the device on which it is mounted.

Note that if you use ANSI "a" characters (which are not alphanumeric) on the volume label on magnetic tape, you must enclose the volume name in quotation marks.

When you use the INITIALIZE command with a magnetic tape volume, INITIALIZE always attempts to read the volume. In some cases, a blank magnetic tape can cause unrecoverable errors, such as the following:

- A fatal system message  
`%INIT-F-VOLINV, volume is not software enabled`
- A runaway magnetic tape  
A runaway magnetic tape occasionally occurs with new magnetic tapes that have never been written to or that have been run through verifying machines.

**Note**

**The only way you can stop a runaway magnetic tape is by taking the magnetic tape drive off line and then putting it back on line.**

If either problem occurs, you can successfully initialize a magnetic tape by repeating the INITIALIZE command from an account that has the VOLPRO and OPER privileges and by specifying the following qualifier in the command:

`/OVERRIDE=(ACCESSIBILITY,EXPIRATION,OWNER)`

This qualifier ensures that the INITIALIZE command will not attempt to verify any labels on the magnetic tape.

### 3.4 Mounting a Volume

Before you can begin processing files or data on your private disk or magnetic tape volume, the volume must be mounted. You can use the DCL command MOUNT to make a disk or magnetic tape volume and the files or data it contains accessible to your process.

When you issue the MOUNT command, the system checks:

- That the device has not been allocated by another user
- That the device protection allows you to allocate the device
- That a volume is physically loaded on the device specified
- That the label on the volume matches the label specified

You can mount a single volume or a volume set. Binding volumes into a volume set allows you to extend the space available for your files by adding volumes to the volume set, rather than defining new volumes. The procedures for creating and mounting disk volume sets and magnetic tape volume sets (as opposed to single volumes) are described in the sections that follow.

Whether you are mounting a disk or a magnetic tape volume, the format for entering the MOUNT command is as follows:

```
$ MOUNT device-name[:][,...] [volume-label[,...]] [logical-name[:]]
```

#### Command Parameters

##### **device-name[:][,...]**

Specifies the physical device name or logical name of the device on which the volume is to be mounted. If you specify more than one device name for a disk or magnetic tape volume set, separate the device names with either commas (,) or plus signs (+).

##### **volume-label[,...]**

Specifies the label on the volume. If you specify more than one volume label, separate the labels with either commas (,) or plus signs (+). The volumes must be in the same volume set. For magnetic tape volumes, the labels must be specified in ascending order according to relative volume number.

The volume-label parameter is not required when you mount a volume with the /FOREIGN or /NOLABEL qualifier or when you specify /OVERRIDE=IDENTIFICATION. To specify a logical name when you enter either of these qualifiers, type any alphanumeric characters in the volume-label parameter position.

### logical-name[:]

Defines a 1 through 255 alphanumeric character string to be associated with the device.

If you do not specify a logical name for a disk drive, the MOUNT command assigns the default logical name DISK\$volume-label to individual disk drives; it assigns the default logical name DISK\$volume-set-name to the device on which the root volume of a disk volume set is mounted. Similarly, if you do not specify a logical name for a magnetic tape drive, the MOUNT command assigns only one logical name, TAPE\$volume-label, to the first magnetic tape device in the list. For a volume set, no logical name is assigned unless you specify one.

The MOUNT command places the name in the job logical name table, unless you specify /GROUP or /SYSTEM. In the latter cases, it places the logical names in the group or system logical name table. You should avoid assigning a logical name that matches the file name of an executable image in SYS\$SYSTEM. Such an assignment will prohibit you from invoking that image.

At many installations, operators perform the physical mounting (and dismounting) of both system and private disk and magnetic tape volumes. Since operators at such installations assist you in your MOUNT requests, you do not need to include the /ASSIST qualifier with the MOUNT command. For example:

```
$ MOUNT DMA1: DISK VOL1
%MOUNT-I-OPRQST, PLEASE MOUNT DEVICE _MARS$DMA1:
```

MOUNT notifies the operator of your mount request and displays a message at your terminal.

After the device has been successfully mounted, you are notified with the following message:

```
%MOUNT-I-MOUNTED, DISK mounted on _DMA1:
```

As an alternative to requesting a specific device, you might want to request a device type. In the following example, MOUNT allocates an available device of the specified type and requests operator assistance in mounting it.

```
$ MOUNT DB: USER_DISK DISK
%MOUNT-I-OPRQST, Please mount volume USER_DISK in device _NODE$DBAO:
%MOUNT-I-MOUNTED, USER_DISK mounted on _NODE$DBAO:
%MOUNT-I-RQSTDON, operator request canceled - mount completed successfully
```

DISK is the logical name created when the RP06 disk unit is allocated. Since the device is allocated to your process, no other user can access the volume. Your access to USER\_DISK is determined by the volume protection code and the volume UIC.

Operator assist messages are sent to all operators enabled to receive TAPE and DISK messages. Thus, if an operator assist is needed for mounting a disk device, a message is sent to disk operators. (See Chapter 7 for more information on operator assistance requests and replies.)

Any operator reply to a MOUNT request is written to SYS\$OUTPUT to be displayed on the user's terminal or written in a batch job log.

If no operator is available (operator is not enabled) to receive and respond to a MOUNT request, a message is displayed to inform you of the situation. A volume placed in the requested drive, needs no additional operator assistance.

Note that you can specify the /NOASSIST qualifier to avoid operator assistance.

The following sections provide examples of mounting volumes; however, they do not all include operator assistance messages. For more examples of operator-assisted mount requests, as well as other operator-oriented functions, refer to Chapter 7.

### 3.4.1 Mounting a Disk Volume

You can use the following procedure to allocate, initialize, and mount a disk volume:

```
$ ALLOCATE DMA2: TEMP
%DCL-I-ALLOC, _MARS$DMA2: allocated
$ INITIALIZE TEMP: BACKUP_FILE
$ MOUNT TEMP: BACKUP_FILE
%MOUNT-I-MOUNTED, BACKUP_FILE mounted on _DMA2:
```

If you wish to mount a foreign disk volume (that is, one in which file structure is other than Files-11), you can use the /FOREIGN qualifier in conjunction with the MOUNT command. The MOUNT/FOREIGN command makes the contents of your volume available to the system, but makes no assumptions concerning its file structure. In the following case, assume that DISK is the logical name assigned to the device name at the time of disk allocation.

```
$ MOUNT/FOREIGN DISK
%MOUNT-I-MOUNTED, BACKUP_FILE      mounted on DISK$DMA2:
```

Note that MOUNT reports a volume label even though the disk is mounted as a foreign device. MOUNT reports the label because the disk has a Files-11 structure; if a disk does not have a recognized file structure, no label is reported.

You need the user privilege VOLPRO to mount a Files-11 structured disk with the /FOREIGN qualifier, unless its owner UIC matched your own.

You must use the `/FOREIGN` qualifier if you wish to use the VAX/VMS Bad Block Locator Utility (BAD) to locate and record bad blocks on your disk volume. BAD is useful for media preparation and is thus distinct from the volume preparation tasks described in this chapter. To invoke BAD, use the DCL command `ANALYZE/MEDIA`. For more information on BAD, see the description in the *VAX/VMS Bad Block Locator Utility Reference Manual*.

In addition to the `/FOREIGN` qualifier, many other qualifiers are supported for the MOUNT command. The `/MOUNT_VERIFICATION` and `/COMMENT` qualifiers are discussed in Chapter 7 of this guide. The `/SYSTEM` qualifier is described in the *VAX/VMS System Manager's Reference Manual*. The `/BIND` qualifier is discussed in the next section of this guide, which describes how to mount disk volume sets. For a complete list of the qualifiers that can be used with the DCL command MOUNT, see the description in the *VAX/VMS Mount Utility Reference Manual*.

### 3.4.2 Mounting a Disk Volume Set

When you mount a disk volume set, the volume label specified in the list must correspond to a device name in the same position in the device name list.

You can bind two or more disk volumes into a volume set. The first volume in the set is called the root volume. Each volume in the set is identified by a relative volume number in the set, where the root volume is always relative volume 1.

A disk volume set has a single directory structure. The master file directory (MFD) for the entire volume set resides on the root volume, which is always the first volume in the set.

When a disk volume set is on line and mounted, all files and directories in the set can be accessed by specifying either of the following:

- Device name of the device on which the root volume is mounted
- Logical name assigned to the volume set when it was mounted

When you mount a disk volume set, each volume label specified in the list must correspond to a device name in the same position in the device name list.

You use the `/BIND` qualifier to create a disk volume set. When you use the MOUNT command with the `/BIND` qualifier to create a disk volume set, the `/BIND` qualifier identifies a volume set by assigning it a volume set name, which applies to all volumes in the set. It also identifies the root volume and creates the directory structure for the volume.

When you create files on a volume set, the file system allocates space for the files anywhere on the set, wherever the most space exists. When existing files on any volume are extended, extension occurs on the same volume, unless the volume is physically full. You can add new volumes to a volume set whenever additional space is needed.

For example, all disk volumes that are mounted on a daily basis can be bound into a volume set. Since this set contains all user file directories, users do not need to specify device names in file specifications to access files on any volume in the volume set. In fact, the physical location of a file is of no concern to users of the system.

### Note

**Do not bind your system disk into a volume set. Volume sets are not supported by VAX/VMS software updates and optional product installation. If certain system files move or extend to other volumes in the set, the system may fail to boot.**

No special privileges are required to use volume sets. However, you must have WRITE access to the index file on all volumes that you are attempting to bind into a volume set; this usually means that you also must have a system UIC, have the user privilege SYSPRV, or be the owner of the volumes.

You can create a disk volume set from newly initialized volumes or you can create a volume set by extending an existing volume that already contains a directory structure and files. You can also add volumes to an existing volume set. The next three sections contain examples of how to create and/or mount each type of disk volume set.

### 3.4.2.1 Creating a Disk Volume Set from New Volumes

This section describes how to create a disk volume set from new disk volumes. The example that follows assumes that there are no files or data on the volumes to be bound.

- 1 Allocate the necessary devices and physically load the volumes.
- 2 Initialize each volume in the set. For example:

```
$ INITIALIZE DB1: PAYVOL1
$ INITIALIZE DB2: PAYVOL2
$ INITIALIZE DB3: PAYVOL3
```

When you initialize volumes for a volume set, you can also use other qualifiers with the INITIALIZE command to define the volume ownership and protection. Protection and ownership information is obtained from the root (first) volume. The protection and ownership of the other volumes is ignored.

- 3 Use the MOUNT/BIND command to create the volume set. For example:

```
$ MOUNT/BIND=MASTER_SET -  
_DB1:, DB2:, DB3: PAYVOL1, PAYVOL2, PAYVOL3
```

This MOUNT/BIND command defines the volume set name, MASTER\_SET, and defines the relative volume numbers of the volumes PAYVOL1, PAYVOL2, and PAYVOL3.

A disk volume set name can have from 1 through 12 alphanumeric characters. The volume set name must be different from all volume labels within the set, and all labels in the set must be unique.

The order of the device names corresponds to the volume labels specified: PAYVOL1 must be physically loaded on DB1, PAYVOL2 on DB2, and PAYVOL3 on DB3.

PAYVOL1, which is listed first in the list of labels, becomes the root volume of the set. The master file directory (MFD) for PAYVOL1 contains the directory structure for the entire volume set.

Note that the MOUNT/BIND command creates the volume set and mounts the volumes. When this command completes successfully, all volumes in the set are ready for use: user file directories can be created.

The /BIND qualifier must be used only once to create the volume set. In subsequent use, the volume set may be mounted with a single MOUNT command. The following example illustrates the use of one MOUNT command to mount a previously created volume set:

```
$ MOUNT DB1,DB2,DB3 PAYVOL1,PAYVOL2,PAYVOL3
```

### 3.4.2.2 Creating a Disk Volume Set from an Existing Volume

This section describes how to create a disk volume set from an existing volume. The example that follows assumes that the volume USERFILES already contains a directory structure and files, and that the volume is currently located on the device DM1.

```
$ INITIALIZE DM2: USERFILES2  
$ MOUNT/BIND=USERS -  
_DM1:, DM2: USERFILES, USERFILES2
```

The initial volume USERFILES must be specified first: it becomes the root volume of the set. When you create a volume set from an existing volume, you must specify that volume first because the file system must build on the existing directory structure.

Note that if you attempt to create a volume set from two or more volumes that already contain files and data, the file system does not issue an error message when you issue the MOUNT/BIND command. However, the volumes are unusable as a volume set because the directory structures are not properly bound.

### 3.4.2.3 Adding Volumes to a Disk Volume Set

This section describes how to add volumes to an existing volume set. The example that follows assumes that the volume set named MASTER\_SET is on line and mounted and has volumes named PAYVOL1, PAYVOL2, and PAYVOL3:

```
$ INITIALIZE DB4: PAYVOL4
$ MOUNT/BIND=MASTER_SET DB4: PAYVOL4
```

This MOUNT command binds the volume PAYVOL4 with the existing volume set and makes the volume ready and available for use. Note that if the volume set MASTER\_PAY was mounted with the /SYSTEM, /GROUP, or /SHARE qualifier, the MOUNT/BIND command that adds a volume to the set must also specify the appropriate qualifier.

When you add a volume to an existing set, the only volume in the set that must be mounted is the root volume, relative volume 1. None of the other volumes need be mounted.

You can also add a volume to a set at the same time you mount the set. The following procedure assumes an existing volume set named MASTER\_SET with volumes named PAYVOL1, PAYVOL2, and PAYVOL3:

```
$ INITIALIZE DB4: PAYVOL4
$ MOUNT/BIND=MASTER_SET DB1:, DB2:, DB3:, DB4: -
_$PAYVOL1, PAYVOL2, PAYVOL3, PAYVOL4/SYSTEM
```

Note that the first device/volume pair listed in the MOUNT/BIND command is the root volume of the set. When you add a volume to a set while mounting the set, you must list the root volume first.

#### Note

**Once a volume is bound into a volume set, it cannot be "unbound."**

You can add volumes to an existing volume set at any time. The maximum number of volumes in a set is 255.

### 3.4.3 Mounting a Magnetic Tape Volume

When mounting a magnetic tape volume, you specify with the MOUNT command any qualifiers you choose, a device name, and optionally a label and logical name.

For a discussion of how to mount magnetic tape volume sets, see Section 3.4.4.

The next two sections describe procedures and commands for mounting single volume ANSI-labeled magnetic tapes.

#### 3.4.3.1 Mounting an ANSI-Labeled Volume

When you use the MOUNT command to mount a magnetic tape volume, VAX/VMS checks to see whether the volume has either a VAX/VMS or non-VAX/VMS ANSI-labeled format. If the format is ANSI labeled, the following are checked:

- The volume identifier field
- The protection on the ANSI-labeled volume as described in Section 2.1.3.

You can mount an ANSI-labeled volume by including the device name and volume identifier as follows (specifying a logical name is optional):

```
$ MOUNT MT: ELAINE ET
%MOUNT-I-OPRQST, please mount volume ELAINE in device $MTA1:
%MOUNT-I-MOUNTED, ELAINE mounted on MTA1:
%MOUNT-I-RQSTDON, operator request canceled -- mount completed successfully
```

MOUNT finds an available MT drive, MTA1:, and requests operator assistance. The message displayed at the user terminal indicates which drive has been selected. At this point you (or the operator) load the magnetic tape on the drive, and the mount operation completes. No operator response is necessary. The display informs you that the volume named ELAINE is mounted on the drive MTA1:. Although not indicated in the message, MOUNT also assigns the logical name ET to the volume ELAINE.

When used with the MOUNT command, the qualifiers described in the next section affect the label format of a volume and/or the magnetic tape file system used to process an ANSI-labeled volume. Unless otherwise noted, you must have VOLPRO privilege to use any of these qualifiers when the volume is a VAX/VMS ANSI-labeled volume containing protection that restricts your process from accessing the volume.

### 3.4.3.2 Using MOUNT Command Qualifiers

This section describes some of the command qualifiers that you can use when mounting a magnetic tape volume. For a complete list of all the command qualifiers supported for the mounting of magnetic tape volumes, see the *VAX/VMS Mount Utility Reference Manual*.

#### **/BLOCKSIZE=n**

You can use the `/BLOCKSIZE=n` qualifier to specify the block size for the magnetic tape. The range of valid values for `n` varies and depends on the density of the volume, whether the data is for input or output, and whether the operation uses RMS.

By default, VAX/VMS writes 2048-byte blocks, which conform to the ANSI standard. Although VAX/VMS allows you to specify a block size larger than 2048 bytes, a larger block size does not conform to ANSI standards.

You must specify `/BLOCKSIZE` when you are mounting volumes that do not support the second file header label as defined in the ANSI standard or support a block size smaller than 2048 bytes. For example, you must specify `/BLOCKSIZE=512` to mount an RT-11 volume.

If you wish to write files on a magnetic tape volume, you can use the `/BLOCKSIZE` qualifier to set the block size to other than 2048. If you wish only to read files from a magnetic tape volume, you do not have to use the `/BLOCKSIZE` qualifier, since VAX/VMS will automatically handle the block size given by the second file header label.

The minimum blocksize for ANSI Standard magnetic tapes is 18, while the maximum is 2048. The minimum blocksize for VAX/VMS magnetic tapes is 14, while the maximum is 65,532.

#### **/LABEL**

You use the `/LABEL` qualifier to indicate that the volume contains standard ANSI labels (this is the default). Privilege is not required to use this qualifier.

#### **/OVERRIDE**

You can use the `/OVERRIDE=(option[,...])` qualifier to inhibit one or more of the access checks performed by MOUNT and the magnetic tape file system. The options are the following:

### ACCESSIBILITY

If the installation allows, this option will override any character in the accessibility field of the volume and file header labels. The necessity of this qualifier is defined by the installation. That is, each installation has the option of specifying a routine that the magnetic tape file system will use to process this field. By default, VAX/VMS provides a routine that checks this field in the following manner.

- If the magnetic tape was created on a version of VAX/VMS that conforms to Version 3 of ANSI, then this option must be used to override any character other than an ASCII space.
- If a VAX/VMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than Version 3, then this option must be used to override any character other than an ASCII 1.

For more information on accessibility-based protection for VAX/VMS-ANSI labeled magnetic tapes, see Chapter 2.

### EXPIRATION

Overrides the expiration dates of a volume and its files. Use this qualifier when the expiration date (in the first file header label) of any file that you want to overwrite has not been reached.

### IDENTIFICATION

Overrides the volume identifier in the volume label. Use this qualifier to mount a volume for which you do not specify the volume identifier. Only the volume identifier field will be overridden. Volume protection, if any, is preserved.

### OWNER\_IDENTIFIER

Overrides the processing of the owner identifier field. You use this option when you need to interchange protected magnetic tapes between VAX/VMS and other DIGITAL operating systems.

### SETID

Prevents MOUNT from checking the file-set identifier in the first file header label of the first file on a continuation volume. Use this qualifier only for ANSI-labeled volumes on which the file-set identifier of the first file on a continuation volume differs from the file-set identifier of the first file of the first volume that was mounted.

### /OWNER\_UIC=uic

You can use the /OWNER\_UIC=uic to override the UIC written in the second volume label and assigns the UIC that you specify to the volume while it is mounted. For magnetic tape volume sets in which a continuation volume is written, the UIC specified at mount time is written to the volume

only if the /PROTECTION qualifier was specified either at mount time or when the volume had been initialized. This does not change the protection on any volumes already created.

You can specify the UIC variable in the format:

[g,m]

- g is either an octal number in the range 0 through 37,776 that denotes the group number or an alphanumeric value (consisting of 1 through 31 characters) that describes the group.
- m is either an octal number in the range 0 through 177,776 that denotes the member number or an alphanumeric value (consisting of 1 through 31 characters) that describes the member.

Either square ([ ]) or angle ( <> ) brackets are required in the UIC specification. For more details on UIC-based protection, see Chapter 2.

### **/CACHE=TAPE\_DATA**

The /CACHE qualifier with the TAPE\_DATA option, enables the write cache for a tape device if the tape controller supports a write cache. /NOCACHE is the default for mounting tape devices. You must specify TAPE\_DATA to enable the write cache. If the tape controller does not support a write cache, the option is ignored. This lets you specify /CACHE=TAPE\_DATA in command procedures that work with a variety of devices and controllers.

This option enables a form of controller-based write-back caching. The write-back caching feature significantly improves the overall performance of streaming tape drives. Under some rare failure conditions, however, some written data can be lost in the cache of the controller. If a failure occurs, the magnetic tape being written becomes seriously flawed or unreadable and you must repeat the entire process used to write to the tape. When a failure occurs you are always notified with an error message.

### **/PROTECTION=code**

The /PROTECTION=code qualifier overrides the VAX/VMS protection written in the second volume label and assigns the protection code that you specify to the volume while it is mounted. For magnetic tape volume sets in which a continuation volume is written, the protection code that you specify will be written to the continuation volume. By default, your process UIC also will be written to the continuation volume unless you explicitly specify an alternate UIC with the /OWNER\_UIC qualifier described above.

Valid protection codes include READ and WRITE access for GROUP and WORLD users; EXECUTE and DELETE access are not applicable to magnetic tape volumes. SYSTEM users and the volume owner always have READ and WRITE access to magnetic tape volumes regardless of the protection code that you specify. Section 2.3 describes access and protection codes.

### **/NOHDR3**

The /NOHDR3 qualifier controls whether special VAX/VMS header labels are written to a volume. Privilege is not required for this qualifier. The default is /HDR3, which allows VAX/VMS header labels to be written to a volume. When the /NOHDR3 qualifier is used, VAX/VMS long file names are truncated to 17 characters. Use the /NOHDR3 qualifier when writing to volumes that will be read by a system other than VAX/VMS, such as the RT-11 system, which does not process all file header labels correctly.

### **/RECORDSIZE=n**

The /RECORDSIZE=n qualifier specifies the number of bytes in each record. This qualifier does not require privilege. Use this qualifier when you mount volumes without the second file header label (such as RT-11 volumes), or when you mount volumes with the /FOREIGN qualifier, to provide RMS with the size of fixed-length records or the maximum size of variable-length records.

The record size must be less than or equal to the specified or default block size. Refer to the /BLOCKSIZE qualifier (described previously) for details. The VAX/VMS operating system does not write records smaller than 14 bytes on output. However, the VAX/VMS Convert Utility, described in the *VAX/VMS Convert and Convert/Reclaim Utility Reference Manual*, allows you to pad and extend the size of records up to and greater than the 14-byte minimum record size output by VAX/VMS.

Two other qualifiers that are important for mounting magnetic tape volumes are /AUTOMATIC and /INITIALIZE. These qualifiers are described in the following sections.

---

#### **3.4.4 Mounting a Magnetic Tape Volume Set**

When mounting a magnetic tape volume set, you can begin by following the procedures described in Section 3.4.3 for mounting a single magnetic tape volume. The number of volume identifiers need not equal the number of device names specified. Thus when you mount a magnetic tape volume set, you can specify more volume identifiers than device names or more device names than volumes.

The number of devices that you specify directly affects the action taken by the magnetic tape file system when processing continuation volumes in a volume set. For example, when the number of devices is greater than the number of volumes, the magnetic tape file system requests a continuation volume to be mounted on the first drive from the list that does not have a volume mounted.

The next two sections describe how to create and mount a magnetic tape volume set. The manner in which continuation volumes are handled by the magnetic tape file system is also described.

#### 3.4.4.1 Creating a Magnetic Tape Volume Set

If you do not create a volume set explicitly, the VAX/VMS system will create one if necessary. If you have not mounted a volume set and a continuation volume is required, the magnetic tape file system will request that a continuation volume be mounted and will implicitly create a volume set. For example, if the magnetic tape file system encounters an EOT mark while writing a volume, it sends a message to the operator console requesting that another volume be mounted. After you mount the next volume, the magnetic tape file system will write the volume and header labels and then will reissue the pending write requests to the continuation volume. The file-set identifier in the first file header label of all files written to the continuation volume will be the file-set identifier of the first file on the first volume. The file-set identifier for VAX/VMS volume sets is always that of the first file of the first volume that was mounted in the set.

To explicitly create a volume set with three volumes, follow this procedure:

- 1 Allocate a drive on which you will load each volume.

```
$ ALLOCATE MTAO:
%DCL-I-ALLOC, _MARS$MTAO: allocated
$ ALLOCATE MTA1:
%DCL-I-ALLOC, _MARS$MTA1: allocated
```

- 2 Initialize the volumes. You should specify the density and the access protection in addition to the device name and the volume identifier in the INITIALIZE commands, as in:

```
$ INITIALIZE/DENSITY=1600/PROTECTION=(G:RW) MTAO: TAPE1
$ INITIALIZE/DENSITY=1600/PROTECTION=(G:RW) MTA1: TAPE2
```

- 3 Mount the volumes. You should include the device name and volume identifier. Specifying a logical name for the volume set is optional.

```
$ MOUNT MTAO:,MTA1: TAPE1,TAPE2,TAPE3 TEST
%MOUNT-I-MOUNTED, TAPE1 mounted on _MTAO:
%MOUNT-I-MOUNTED, TAPE2 mounted on _MTA1:
```

VAX/VMS not only confirms which volumes have been mounted but also indicates on which drive each volume has been mounted.

VAX/VMS mounts and verifies only the volumes that are physically loaded on the drives at mount time. However, the volume identifiers of additional volumes that you specify will not be verified until the volumes are accessed.

You can check the densities, volume labels, UICs, and relative volume numbers of the volumes that are mounted on drives. To do so, specify the `SHOW DEVICES/FULL` command. If you specify a generic device code for the magnetic tape drives, such as `MT`, information for all the drives of that type that are configured in the system will be displayed. To display information for a volume mounted on a specific drive, specify the physical device code, consisting of the generic device code, the controller designation, and the unit number followed by a colon. For more information on the `SHOW DEVICES` command, including examples of displays returned by the `SHOW DEVICE/FULL` command, see Chapter 4. (See also the *VAX/VMS DCL Dictionary*.)

#### **3.4.4.2 Mounting Continuation Volumes in a Volume Set**

When mounting a magnetic tape volume set, you can follow the general procedures described in the previous section for creating a magnetic tape volume set. Once the volume set has been created, however, there is no need to initialize the volumes in the set when you mount the volume set.

You need not allocate a drive for each volume in the volume set. The magnetic tape file system will request that volumes be switched to appropriate drives when continuation volumes are required.

VAX/VMS stores, but cannot verify, the volume identifiers of volumes that you specify but do not physically mount on drives at mount time. VAX/VMS later verifies the volume identifiers of such volumes when the volumes are accessed.

The VAX/VMS operating system supports the continuous processing of mounted volumes in a magnetic tape volume set through automatic volume switching. To do this, the magnetic tape file system uses automatic volume recognition (AVR) and automatic volume labeling (AVL).

To take advantage of this automatic volume switching capability, you must have more than one magnetic tape drive allocated to your volume set. If you have two or more magnetic tape drives allocated to a volume set, the magnetic tape file system will switch volumes for you automatically by sequentially selecting the next magnetic tape drive allocated to the volume set. The magnetic tape file system will expect the next volume in the volume set to be loaded on that drive.

If the file system is writing to the volume set, then it will create a label for the magnetic tape and initialize the magnetic tape with that label and with the protection characteristics set for the first of the volume set. If the magnetic tape file system is reading the volume set, it will generate the label and will try to mount the next magnetic tape with that label. If the drive has no magnetic tape loaded on it or if the drive has the wrong magnetic tape loaded on it, then the magnetic tape file system will send a message to the operator console notifying the operator either to mount a magnetic tape or mount the correct magnetic tape.

Before processing continuation volumes, the magnetic tape file system processes the protection on that volume (as described in Section 2.1.3). If the magnetic tape file system determines that the user does not have access to the volume, then a message is sent to the operator to take some action.

The label generated fills the six-character volume identifier field. The first four characters of the field contain the first four characters of the label specified for the previous volume in the volume set. (If the label is less than four characters, the volume identifier field will be padded with underscores.) The fifth and sixth characters contain the relative volume number for that reel in the volume set. Note that this allows VAX/VMS to generate only 99 unique labels for a given volume set.

With automatic volume switching enabled, the operator can load a magnetic tape on the next drive allocated to the magnetic tape volume set anytime before the volume being processed reaches the EOT mark. The magnetic tape file system will mount and initialize (if INITIALIZE was specified originally) the next magnetic tape in the volume set and then notify the operator that the switch has occurred.

In the following example, the volume with the identifier TAPE is mounted on MTA0:

```
$ MOUNT MTA0:,MTA1:,MTA2: TAPE
```

Continuation volumes for this set should be loaded on the magnetic tape drives in the following order: MTA1:, MTA2:, MTA0:, MTA1:, MTA2:, and so forth.

You can explicitly override automatic volume switching by specifying the /NOAUTOMATIC qualifier when mounting a magnetic tape volume. The default is /AUTOMATIC. If you allocate only drive to the magnetic tape volume set, you will implicitly disable automatic volume switching.

To ensure that any volume added to the magnetic tape volume set will be initialized prior to being written to, you can mount the volume with the /INITIALIZE=CONTINUATION qualifier. The default is /NOINITIALIZE.

The next example demonstrates the use of the /INITIALIZE=CONTINUATION qualifier for mounting volume sets. It also shows how volume identifiers are generated for continuation volumes.

```
* INITIALIZE MTA0: MAIN
* MOUNT/OVERRIDE=IDENTIFIER/INITIALIZE=CONTINUATION MTA0:,MTA1:
```

The volume labeled MAIN will be mounted on MTA0:. The second volume in the set will receive the volume identifier MAIN02 and will be mounted on MTA1:. The third volume in the set will receive the volume identifier MAIN03 and will be mounted on MTA0:.

```
* MOUNT MTA0:,MTA1: SUN
```

In this example, the first volume in the set is labeled SUN and is mounted on MTA0:. The second volume will receive the identifier SUN\_02 and be mounted on MTA1:. The third volume will receive the identifier SUN\_03 and be mounted on MTA0:.

The next example illustrates a continuation volume with two volume identifiers.

```
* MOUNT MTA0:,MTA1: SUN,MOON
```

In this case, SUN and MOON will be mounted on MTA0: and MTA1: respectively. If a third volume is added to the set, it will be given the identifier MOON03 and be mounted on MTA0:.

For a description of the operator functions in handling user requests to mount magnetic tape volume sets and process continuation volumes, see Chapter 7.

---

### 3.5 Dismounting a Volume

When you have finished processing the files or data on your disk or magnetic tape volume, you can explicitly dismount the volume. You can use the DISMOUNT command to explicitly dismount a single volume or an entire volume set.

If you explicitly dismount a single volume in a volume set, VAX/VMS will dismount all the volumes in the set. For disk volume sets, however, it is possible to explicitly dismount a single volume in the volume set without dismounting the entire set. To do this, you must use the /UNIT qualifier.

When you issue the DCL command DISMOUNT, the volume is automatically unloaded from the drive. You can override this automatic unloading of your volume by specifying the /NOUNLOAD qualifier with the DISMOUNT command.

Even when you specify the /NOUNLOAD qualifier with the DISMOUNT command, your volume will still be logically dismounted from the drive. However, the volume will remain physically loaded on the drive. If you use the /NOUNLOAD qualifier to dismount a magnetic tape volume, the volume will remain loaded on the magnetic tape drive and the magnetic tape reel will be rewound to the BOT mark.

If you plan to remount or reinitialize a volume that you are dismounting, you can save time and eliminate unnecessary handling of that volume by using the /NOUNLOAD qualifier with the DISMOUNT command.

The following examples show how to use the DISMOUNT command. The first example employs the /NOUNLOAD qualifier.

```
$ DISMOUNT/NOUNLOAD MTA1:
$
```

In this example, the magnetic tape volume will be logically dismounted and remain loaded on drive MTA1:. Also, the magnetic tape reel will be rewound to the BOT mark. The VAX/VMS system returns you to DCL level.

The DISMOUNT command is also used to dismount foreign volumes. The following command dismounts a volume that had been mounted with the /FOREIGN qualifier on DBA0:.

```
$ DISMOUNT DBA0:
$
```

In this example, the volume that had been mounted with the /FOREIGN qualifier on DBA0: will be dismounted and automatically unloaded. The VAX/VMS system returns you to DCL level.

As mentioned previously, you use the DISMOUNT command to dismount an entire volume set. If you explicitly dismount any volume in a disk or magnetic tape volume set, the entire volume set will be dismounted. For example, if you had a volume set that consisted of DBA3: and DBA4: and you entered the following command, the entire volume set would be dismounted:

```
$ DISMOUNT DBA3:
```

It is possible to dismount individual disk volumes in a volume set without dismounting the entire disk volume set. You can do this by using the /UNIT qualifier with the DISMOUNT command. The /UNIT qualifier specifies that only the disk volume on the specified device is dismounted. For example:

```
$ DISMOUNT/UNIT DBA4:
```

This command specifies that only DBA4: (included in the disk volume set mentioned above) is to be dismounted.

You should always explicitly dismount a volume or volume set with the DISMOUNT command, or a command procedure that contains that command, before physically unloading that volume.

A volume will be dismounted and unloaded automatically if you log out of the job from which you had mounted the volume. If the system crashes, however, the drive is not automatically dismounted.

Note that data corruption can occur if a volume has not been explicitly dismounted and the system fails. For magnetic tape volumes, data corruption can occur if you unload a volume that contains an open file for which file trailer labels had not been written. When you remount the volume and attempt to access the file without file trailer labels, you will receive the following error message:

%MTAACP-magnetic tape position lost

You will be able to access all the files (on that magnetic tape volume) preceding the file whose file trailer labels had not been written. However, you will not be able to access the file without file trailer labels.

Note that the dismount of a volume is done by the file system and is not complete until all the open files on the volume have been closed. Thus, a substantial amount of time can pass between the time you issue the DISMOUNT command and the completion of the dismount. Always wait for the drive to unload before you remove the volume. (You can verify that the dismount is complete by using the DCL command SHOW DEVICES.)

If the device you are dismounting was allocated with an ALLOCATE command, it remains allocated after it is dismounted with the DISMOUNT command. If the device was implicitly allocated by the MOUNT command, the DISMOUNT command deallocates it.

For more information on the DISMOUNT command, see the *VAX/VMS DCL Dictionary*.

---

### 3.6 Deallocating Drives

The process of allocation reserves a device for exclusive use by your process. The device will remain allocated to your process until you explicitly deallocate it or until you log out from your process. Once you have allocated the device, other users cannot access that device until you explicitly deallocate it or log out.

You can use the DCL command DEALLOCATE to deallocate explicitly a disk drive or magnetic tape drive that has been allocated to your process. A complement to the ALLOCATE command, the DEALLOCATE command logically disconnects a drive from your process and returns it to the pool of devices.

The following examples show how to explicitly deallocate a magnetic tape drive and a disk drive.

```
$ DEALLOCATE MTA1:  
$
```

In this example, the DEALLOCATE command logically disconnects magnetic tape drive MTA1: from your process. The VAX/VMS system returns you to DCL level.

```
$ DEALLOCATE DJA0:  
$
```

In this example, the DEALLOCATE command logically disconnects disk drive DJA0: from your process. The VAX/VMS system returns you to the DCL command level.

Since logging out of a process from which drives have been allocated will automatically deallocate all explicitly and implicitly allocated drives, you do not have to explicitly deallocate a disk or magnetic tape drive that has been allocated to your process. However, it is a good practice to use the DEALLOCATE command (or a command procedure containing this command) to explicitly deallocate all the drives you allocated with the ALLOCATE command.

---

### 3.7 Using Command Procedures to Set Up Disks and Magnetic Tapes

Since private disk and magnetic tape volumes frequently must be set up before you can perform operations on them, you may want to design command procedures to facilitate the set-up procedures. This section contains examples of command procedures that can be used to set up disk and magnetic tape volumes for routine operations.

You can tailor command procedures to meet the needs of your own set-up tasks. The command procedure examples in this section, although general in nature, can serve as guiding strategies for you.

The next two sections contain examples of command procedures that are designed to set up disk and magnetic tape volumes for routine processing.

### 3.7.1 Designing Command Procedures to Set Up Disk Volumes

The following command procedure is designed to allocate, initialize, and mount a disk volume.

You can use a text editor, such as EDT or EVE to create a file to contain your command procedure. Assume that a file named SETUP.COM has been created and that it contains a very basic command procedure, which, when executed, allocates and mounts a disk.

You can construct this command procedure by entering the following text:

```
$ ! Place a disk in the drive
$ IF P1 .EQS. "" THEN INQUIRE P1 "enter device name"
$ IF P2 .EQS. "" THEN INQUIRE P2 "enter volume label"
$ IF P3 .EQS. "" THEN INQUIRE P3 "enter logical name"
$ ALLOCATE 'P1'
$ MOUNT 'P1' 'P2' 'P3'
```

This command procedure, although very simple, accomplishes the task of allocating and mounting a disk, each time it is executed. It is designed to prompt you for the device name, volume label, and logical name of the disk device that you want to allocate and mount. By assigning logical names to your disks, you can use this command procedure to allocate and mount devices over and over again.

You can take further advantage of the power of a command procedure by including a few additional tasks as well. For example, you could design the command procedure to deallocate and dismount the disk. The command procedure example used to set up a magnetic tape (described in the next section) takes advantage of some of these options.

You execute the above command procedure by entering the following command:

```
$ @SETUP
```

You can also write command procedures to mount a volume from a batch job. By using logical names to refer to devices and files, you can use the same command procedures without modification each time you want to access a volume.

For example, if you use the same RK07 disk pack to back up your files on a weekly basis, you can submit as a batch job a command procedure such as the following:

```
$ MOUNT DM: BACK_UP_GMB RK
$ BACKUP/REPLACE *.* RK:*.
$ DIRECTORY/FULL/OUTPUT=BACKUP.LOG RK:
$ DISMOUNT RK:
```

In this command procedure, the MOUNT command will find and allocate a device of the type DM: and create a logical name RK. The MOUNT command substitutes the equivalence name in the message displayed at the operator console.

When the MOUNT command notifies the operator to mount the correct volume, the job waits until the operator responds.

### 3.7.2 Designing Command Procedures to Set up Magnetic Tape Volumes

The command procedure below, which is more sophisticated and detailed than the previous example, is designed to set up a magnetic tape for processing. The ALLOCATE and MOUNT/FOREIGN commands are included in this command procedure.

Using a text editor, you can construct a command procedure in the following way:

```
$ ! First mount the tape on the drive
$ ON CONTROL_Y THEN GOTO EXIT
$ ON ERROR THEN GOTO EXIT
$ WRITE SYS$OUTPUT "Welcome to FETCH."
$ WRITE SYS$OUTPUT " "

$ L1: INQUIRE/NOPUNC PHYS "Have you placed the volume in the drive? "
$ IF .NOT. PHYS THEN GOTO L1
$ INQUIRE/NOPUNC DRIVE "Which drive is the volume mounted on? "
$ DRIVE = DRIVE - ":"
$ ALLOCATE 'DRIVE'
$ MOUNT/FOREIGN 'DRIVE'
$ ON ERROR THEN GOTO COMMAND_LOOP
$ !
$ COMMAND_LOOP: INQUIRE/NOPUNC OPTION "FETCH> "
$ IF OPTION .EQS. "DIR" THEN GOTO DIR
$ IF OPTION .EQS. "EXIT" THEN GOTO EXIT
$ IF OPTION .EQS. "FETCH" THEN GOTO FETCH
$ IF OPTION .EQS. "HELP" THEN GOTO HELP
$ IF OPTION .EQS. "LIST" THEN GOTO LIST
$ GOTO COMMAND_LOOP
$ !
$ DIR: INQUIRE SPEC "Filespec"
$ DIR 'SPEC'
$ GOTO COMMAND_LOOP
```

```

$ HELP:
$ WRITE SYS$OUTPUT "Enter any of the following commands at the prompt:"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "DIR          (To search for a file)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "EXIT        (To exit this program)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "FETCH      (To perform a BACKUP RESTORE operation)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "HELP      (To read this text)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "LIST      (To perform a BACKUP LIST operation)"
$ GOTO COMMAND_LOOP
$ !
$ FETCH: INQUIRE FILE "Filespec"
$ INQUIRE SAVESET "Save set name"
$ LINE := BACKUP/LOG 'DRIVE': 'SAVESET'/SELECT='FILE'
$ INQUIRE EXCLUDE "Enter any filespecs you want excluded"
$ IF EXCLUDE .EQS. "" THEN GOTO L2
$ LINE := 'LINE'/EXCLUDE=('EXCLUDE')
$ !
$ L2: INQUIRE/NOPUNC TO "Where do you want the file(s)? (<RET> for current directory)"
$ IF TO .EQS. "" THEN GOTO REPLACE
$ LINE := 'LINE' 'TO'
$ GOTO L3
$ REPLACE: LINE := 'LINE' []
$ !
$ L3: INQUIRE/NOPUNC NEW "Create a new version if file already exists?"
$ IF .NOT. NEW THEN GOTO NOT
$ LINE := 'LINE'/NEW_VERSION
$ !
$ NOT: LINE := 'LINE'/OWNER_UIC=ORIGINAL
$ LINE
$ GOTO COMMAND_LOOP
$ !
$ LIST: INQUIRE SPEC "Filespec"
$ INQUIRE SAVESET "Save set name"
$ INQUIRE/NOPUNC OUTPUT "What do you want to call the list file? (<RET> for SYS$OUTPUT )"
$ IF OUTPUT .EQS. "" THEN GOTO NOOUT
$ LINE := BACKUP/LIST='OUTPUT' 'DRIVE': 'SAVESET'/SELECT=('SPEC')
$ GOTO L4
$ NOOUT: LINE := BACKUP/LIST 'DRIVE': 'SAVESET'/SELECT=('SPEC')
$ !
$ L4: INQUIRE EXCLUDE "Enter any filespecs you want excluded"
$ IF EXCLUDE .EQS. "" THEN GOT L5
$ LINE := 'LINE'/EXCLUDE=('EXCLUDE')
$ !
$ L5: LINE
$ GOTO COMMAND_LOOP
$ !
$ EXIT:
$ DISMOUNT 'DRIVE'
$ DEALLOCATE 'DRIVE'

```

Assume that the above command procedure is contained in a file named `FETCH.COM`. You would execute this command procedure by entering the following:

`@FETCH`

This command procedure is more complex than the two previous ones, which were both used to set up a disk. Like the former command procedures, the procedure contained in `FETCH.COM` also accomplishes the basic task of allocating the device and mounting the volume.

In addition to its allocating and mounting functions, however, the command procedure contained in `FETCH.COM` is designed to prompt you for input. For example, it specifically asks you if the magnetic tape is on the drive.

Note that this command procedure is also designed to do a `BACKUP` restore operation. It prompts you for specific options on the restore operation.

Finally, this command procedure explicitly dismounts your magnetic tape volume and deallocates the drive, after your task has completed.

# 4

## Manipulating Files on Disks and Magnetic Tapes

This chapter describes how to manipulate disk and magnetic tape files by using the DIGITAL Command Language (DCL). In particular, it describes how you can use DCL commands to perform the following tasks:

- Retrieve disk and magnetic tape file information
- Modify disk and magnetic tape file characteristics
- Access files residing on disk and magnetic tape volumes

It also contains examples of DCL command procedures designed to access files for create, read, and write operations.

### 4.1 Using DCL to Manipulate Files

In VAX/VMS, you can use the DCL command language to manipulate disk and magnetic tape files. One advantage of manipulating files by way of DCL is that you can do so without ever leaving the command level.

DCL enables you to manipulate files in the following ways. You can use the appropriate DCL SHOW commands to retrieve disk and magnetic tape file information such as device and protection characteristics, and display this information on your terminal. You can use the appropriate DCL SET commands to modify disk and magnetic tape file characteristics, such as protection or UIC information. You can also use the DCL command language to access disk and magnetic tape files for read and/or write operations.

In addition to manipulating files through DCL, you can write user programs to assist you in routine file-manipulation tasks. You can write these programs in either VAX MACRO or in one of the higher-level languages supported by VAX/VMS. If you wish to manipulate individual records within files — that is, if you wish to access files at the record level — you should write programs that include RMS facilities. Examples of the RMS facilities that you can use to manipulate files at the record level are included in the *VAX Record Management Services Reference Manual*.

This chapter focuses on the ways in which you can use DCL to manipulate disk and magnetic tape files at the file level. Note the following restrictions on the use of DCL commands for manipulating files on disk and magnetic tape volumes:

- The SUBMIT command cannot access files on allocated devices. You can submit files on private disk volumes if you mount the volume as a shareable volume. To submit a file on a magnetic tape volume, you must copy the file first to a shared disk volume and then submit it from the disk.
- You can print a file from a privately owned volume. Note, however, that the volume containing the file you wish to print must remain mounted until after the file has completed printing. If you do not want to wait until the printing completes, you can copy the file directly to the line printer, using the DCL command COPY:  

```
$ COPY MYPHILE.DAT LPAO:
```
- Most DCL commands require file-structured devices. For a list of those commands that do not require file-structured devices, see the *VAX/VMS DCL Dictionary*.
- You can execute a command procedure that resides on a magnetic tape volume as long as the procedure does not invoke other procedures and does not issue any GOTO commands referring to labels in the procedure that precede the GOTO command. In this case, it would be better to copy the command procedure from the magnetic tape volume to a local disk, from which you can then invoke the command procedure.

Note that you cannot use DCL commands to read or write files that are not in the standard formats supported by VAX/VMS (these formats are described in greater detail in the *Guide to Using DCL and Command Procedures on VAX/VMS*).

---

### 4.2 Using DCL to Retrieve File Information

The DCL command language provides commands that enable you to retrieve information about disk and magnetic tape files, volumes, and devices. You can use the following DCL commands to retrieve such information:

- DIRECTORY
- SHOW ACL
- SHOW DEVICES
- SHOW MAGTAPE
- SHOW PROTECTION
- SHOW QUOTA

See the *VAX/VMS DCL Dictionary* for a complete list of the command qualifiers and parameters applicable to each of these DCL commands.

## 4.2.1 Retrieving Directory Information

You can use the DCL command DIRECTORY to retrieve information about a file or a group of files residing on a disk or magnetic tape volume.

You can use the DIRECTORY command to list the names of all of the files in a particular directory, or you can use it in conjunction with a file specification to list the names of specific files in a given directory. When you include certain command qualifiers along with the DIRECTORY command, you can retrieve information in addition to the names of the files.

The examples that follow illustrate three cases of retrieving information from the [MALCOLM] directory, which resides on a disk whose logical name is DISK\$DOCUMENT. Note that the asterisk wildcard character is used in the file specifications of two of the disk directory examples.

### EXAMPLES

- 1 \$ DIRECTORY AVERAGE.\*  
Directory DISK\$DOCUMENT: [MALCOLM]  
AVERAGE.EXE;6      AVERAGE.FOR;6      AVERAGE.LIS;4      AVERAGE.OBJ;12  
Total of 4 files.
  
- 2 \$ DIRECTORY/SIZE=USED/DATE=CREATED/VERSIONS=1/PROTECTION AVERAGE  
Directory DISK\$DOCUMENT: [MALCOLM]  
AVERAGE.EXE;6      6      10-APR-1986 15:43 (RWED,RWED,RWED,RE)  
AVERAGE.FOR;6      2      2-APR-1986 10:29 (RWED,RWED,RWED,RE)  
AVERAGE.LIS;4      5      9-APR-1986 16:27 (RWED,RWED,RWED,RE)  
AVERAGE.OBJ;6      2      9-APR-1986 16:27 (RWED,RWED,RWED,RE)  
Total of 4 files, 15 blocks.
  
- 3 \$ DIRECTORY/FULL/VERSIONS=1 [MALCOLM...]AVERAGE.EXE  
Directory DISK\$DOCUMENT: [MALCOLM]  
AVERAGE.EXE;6      File ID: (4098,149,0)  
Size:      36/36      Owner: [DOCUMENTATION,MALCOLM]  
Created: 27-JUN-1986 12:22      Revised: 27-JUN-1986 12:22 (2)  
Expires: <None specified>      Backup: 3-JUL-1986 22:03  
File organization: Sequential  
File attributes:      Allocation: 36, Extend: 36, Global buffer count: 0  
                            No version limit  
Record format:      Variable length, maximum 255 bytes  
Record attributes: Carriage return carriage control  
File protection:      System:RWED, Owner:RWED, Group:RE, World:  
Access Cntrl List: None  
Total of 1 file, 36/36 blocks.

```
Directory DISK$DOCUMENT: [MALCOLM.TEST]
AVERAGE.EXE;1          File ID: (7714,29,0)
Size: 36/36            Owner: [DOCUMENTATION,MALCOLM]
Created: 15-APR-1986 10:12 Revised: 15-APR-1986 10:12 (1)
Expires: <None specified> Backup: 15-APR-1986 22:41
File organization: Sequential
File attributes: Allocation: 36, Extend: 36, Global buffer count: 0
                  No version limit
Record format: Variable length, maximum 255 bytes
Record attributes: Carriage return carriage control
File protection: System:RWED, Owner:RWED, Group:RE, World:
Access Cntrl List: None
Total of 1 file, 36/36 blocks.
Grand total of 2 directories, 2 files, 72/72 blocks.
```

Directory structures do not apply to magnetic tape volumes. However, you can use the DIRECTORY command to search for files on magnetic tape volumes.

This use of the DIRECTORY command is also discussed in Section 4.4.2, which describes how to access magnetic tape files for read and write operations.

If you want to find the names of all files on a magnetic tape volume mounted on MTA2:, you would enter the following command:

```
$ DIRECTORY MTA2:
```

This directory command lists the file names and file types of all files on the magnetic tape.

As in the case of the disk examples included above, you can use wildcard characters in directory specifications for magnetic tapes. For example, if you enter the command:

```
$ DIRECTORY MFA0:*. *;*
```

VAX/VMS will search the entire volume set. Both ANSI and VAX/VMS file names will be returned. (The difference between these two types of magnetic tape files is described in Section 4.4.2).

---

### 4.2.2 Retrieving Device Information

You can use the DCL command SHOW DEVICES to retrieve information about the availability of devices on your system.

When you issue the SHOW DEVICES command without specifying a device or using any qualifier, information about all devices on the system is displayed. If you specify a device name, SHOW DEVICES displays information about that device. If you use certain qualifiers with SHOW DEVICES, information is displayed about those devices that currently have volumes mounted and/or that have been allocated to processes.

The device name displayed by the system uses the format *ddcu*, where *dd* is the device code, *c* is the control, and *u* is the unit. If the system is part of a VAXcluster, the device name will include the node name using the format *node\$ddcu*, where *node* refers to the node name of the system that the device resides on, *dd* refers to the device type, *c* refers to the controller designation, and *u* refers to the unit number. See the *Guide to VAXclusters* for a complete description of the device name format on clusters.

The examples that follow include three instances of how the SHOW DEVICES command can be used.

## EXAMPLES

1 \$ SHOW DEVICES

Device Name	Device Status	Err. Count	Volume Label	Free Blocks	Trans Count	Mount Count
DBAO:	Online mnt	0	VMS	47088	115	1
DBA1:	Online mnt	0	USERPACK1	45216	2	1
DBA2:	Online mnt	3	DOCUMENT	8068	20	1
DBA5:	Online mnt	0	MASTERP	28668	1	1
DBA6:	Online	0				
DBA7:	Online mnt	0	PROJECT	110547	1	1
DMAO:	Online	0				
DLAO:	Online	0				
DYAO:	Online	0				
DYA1:	Online	0				
DRA3:	Online mnt	0	RES26APR	29317	1	1
MFAO:	Online	8				
MFA1:	Online	1				
MTAO:	Online mnt	9	BACKUP	453	1	1
MTA1:	Online	0				

The SHOW DEVICES command displays the following information for each device on the system:

- Device name
- Device status and characteristics. (Status indicates whether the device is on line; characteristics indicate whether the device is allocated, spooled, has a volume mounted on it, and if the volume is mounted foreign.)
- Error count
- Volume labels

## Manipulating Files on Disks and Magnetic Tapes

- Number of free blocks on the volume (disk only)
- Transaction count
- Number of mount requests issued for the volume (disk devices only)

### 2 \$ SHOW DEVICES/FULL DMA0

Disk \$1\$DMA0: (NODE1), device type RK07, is online, allocated, served to the cluster, error logging enabled.

Error count	2	Operations completed	4527
Owner process	"SMITH"	Owner UIC	[0,0]
Owner process ID	24400133	Dev Prot	S:RWED,O:RWED,G:RWED,W:RWED
Reference count	1	Default buffer size	512
Allocation class	1		

The SHOW DEVICES command requests a full listing of the status of the RK07 device DMA0. The device is located on NODE1 in a VAXcluster.

### 3 \$ SHOW DEVICES/FULL NODE2\$

Disk \$1\$DBA0: (NODE2), device type RP05, is online, mounted, file-oriented device, shareable, served to cluster via MSCP Server, error logging is enabled.

Error count	0	Operations completed	120
Owner process	" "	Owner UIC	[303,2]
Owner process ID	00000000	Dev Prot	S:RWED,O:RWED,G:RWED,W:RWED
Reference count	1	Default buffer size	512
Total blocks	171798	Sectors per track	22
Total cylinders	411	Tracks per cylinder	19
Allocation class	1		
Volume label	"HIGHNOON"	Relative volume number	0
Cluster size	3	Transaction count	1
Free blocks	94425	Maximum files allowed	21474
Extend quantity	5	Mount count	8
Mount status	System	Cache name	"_\$255\$DUA8:XQPCACHE"
Extent cache size	64	Maximum blocks in extent cache	9442
File ID cache size	64	Blocks currently in extent cache	0
Quota cache size	0	Maximum buffers in FCP cache	421

Volume status: subject to mount verification, file high-water marking, write-through caching enabled.

Volume is also mounted on NODE1, NODE4, NODE3.

Disk NODE2\$DBC1:, device type RP06, is online, error logging enabled.

Error count	0	Operations completed	0
Owner process	" "	Owner UIC	[0,0]
Owner process ID	00000000	Dev Prot	S:RWED,O:RWED,G:RWED,W:RWED
Reference count	0	Default buffer size	512
Host name	"NODE2"	Host type, available	V780, yes

```

Disk NODE2$DMA0:, device type RK07, is online, error logging enabled.
Error count          0      Operations completed          0
Owner process        ""      Owner UIC                    [0,0]
Owner process ID     00000000 Dev Prot   S:RWED,O:RWED,G:RWED,W:RWED
Reference count      0      Default buffer size      512
Host name            "NODE2" Host type, available      V780, yes

```

In this example, you have requested a full display of information about each device on NODE2 on the VAXcluster. Information is shown here only for the first three devices: a mounted device and two that are not mounted.

## 4.2.3 Retrieving Magnetic Tape Device Information

You can use the DCL command SHOW MAGTAPE to display the current characteristics and status of a specified magnetic tape device.

Although you can use the SHOW DEVICES command to find available magnetic drives on your system, the SHOW MAGTAPE enables you to retrieve additional information about the characteristics of a particular magnetic tape device.

When you enter the SHOW MAGTAPE command at your terminal, you will receive the following prompt:

\_Device:

You must then specify the name of the magnetic tape device for which you want to display the characteristics and status.

The following example illustrates how the SHOW MAGTAPE command is used to retrieve information about MTA0::

### EXAMPLE

```

$ SHOW MAGTAPE MTA0:
MTA0:, device type TU77, is on line, error logging enabled
Error count          0      Operations completed          0
Owner process        ""      Owner UIC                    [0,0]
Owner process ID     00000000 Dev Prot S:RWED, O:RWED, G:RWED, W:RWED
Reference Count      0      Default buffer size      2048
Density              800      Format                    Normal-11
Odd Parity

```

This SHOW MAGTAPE command displays the characteristics of the device MTA0:. Among other characteristics, it displays the device type, density, and format.

---

#### 4.2.4 Retrieving Disk File Protection Information

You can use the DCL commands DIRECTORY/ACL, SHOW ACL, or SHOW PROTECTION to display the current process default protection. This protection is applied to files created during your terminal session, or batch jobs, where defaults from directories or previously existing versions are not available.

You can change the default protection at any time with the SET PROTECTION command. (The SET ACL and SET PROTECTION commands are discussed in Section 4.3, which describes how to modify file characteristics.)

This section is not applicable to magnetic tapes. Although you can use the SHOW/SET PROTECTION commands to SHOW/SET the default protection of magnetic tapes, the protection will not get written to the magnetic tape volume unless you specify the /PROTECTION= qualifier with the INITIALIZE command when you are initializing the magnetic tape volume. See the description of initializing magnetic tape volumes in Chapter 3 of this guide.

The following example illustrates how the SHOW PROTECTION command can be used in conjunction with the SET PROTECTION command to display and modify the protection characteristics of a disk file.

---

#### EXAMPLE

```
$ SHOW PROTECTION
SYSTEM=RWED, OWNER=RWED, GROUP=RE, WORLD=NO ACCESS
$ SET PROTECTION=(GROUP:RWED,WORLD:RE)/DEFAULT
$ SHOW PROTECTION
SYSTEM=RWED, OWNER=RWED, GROUP=RWED, WORLD=RE
```

This SHOW PROTECTION command requests a display of the current protection defaults. The SET PROTECTION/DEFAULT command is then used to change the file access allowed to other users in the same group as well as to miscellaneous system users.

#### 4.2.5 Retrieving Disk Quota Information

It is frequently important to limit the amount of disk space certain users consume. The VAX/VMS Disk Quota Utility (DISKQUOTA) provides the system manager with this capability. Users who have READ access (R) to the quota file can use the SHOW QUOTA command to determine how much disk space any user on the system has been allocated. Users who do not have READ access to the quota file can use the SHOW QUOTA command to determine their own allotments.

You can manage the checking on a per-volume basis at mount time by using the MOUNT qualifier /[NO]QUOTA. (You can also use DISKQUOTA to enable and disable amounts of disk space.) You must have the VOLPRO user privilege or your UIC must match the UIC written on the volume in order to use the MOUNT command qualifier /QUOTA.

You can use the DCL command SHOW QUOTA to find out whether a quota exists for any specific user on a specific disk. The display that results from the SHOW QUOTA command gives the quotas used, authorized, and available.

A user can have a quota for any disk volume on the system. In some cases, the user is permitted a certain authorized limit plus an overdraft limit. Generally, only the authorized limit applies. However, certain system programs, such as editors, can employ the overdraft feature when the authorized limit is exceeded.

If you should run out of disk space during the creation of a file, you receive a system message. If you find you cannot obtain sufficient space by purging or deleting unnecessary files, you may need to contact the system manager to increase your disk quota. If you attempt to write a file to a spooled printer, you must have WRITE access and have sufficient quota on the disk that is associated with that printer.

The examples that follow contain two instances of the SHOW QUOTA command.

#### EXAMPLES

```
1 $ SHOW QUOTA
User [DOCUMENTATION,MALCOLM] has 2780 blocks used, 7220 available,
of 10000 authorized and permitted overdraft of 500 blocks on DISK$
```

The SHOW QUOTA command displays the amount of disk space authorized, used, and still available on the current default disk for the present user. The permitted overdraft in this example is 500 blocks.

2 \$ SHOW QUOTA /USER=[DOCUMENTATION,JONES]/DISK=XXX1:  
%SYSTEM-F-NODISKQUOTA, no disk quota entry for this UIC

This SHOW QUOTA command displays the fact that the user with UIC [DOCUMENTATION,JONES] has no disk quota allocation on device XXX1.

3 \$ SHOW QUOTA /USER=[DOCUMENTATION,ELAINE]  
User [DOCUMENTATION,ELAINE] has 27305 blocks used, 2305 OVERDRAWN,  
of 25000 authorized and permitted overdraft of 4000 blocks on DISK\$

This SHOW QUOTA command illustrates a user with an overdrawn quota.

---

### 4.3 Using DCL to Modify File Characteristics

DCL provides commands that enable you to establish and modify the characteristics of files residing on disk and magnetic tape volumes. To establish or modify file characteristics, you can use the following commands:

- SET ACL
- SET DIRECTORY (disk only)
- SET FILE (disk only)
- SET MAGTAPE (magnetic tape only)
- SET PROTECTION (disk only)
- SET VOLUME (disk only)

Each of these commands is discussed in detail in the *Guide to Using DCL and Command Procedures on VAX/VMS*. For a complete list of the command qualifiers and parameters applicable to each of these DCL commands, see the *VAX/VMS DCL Dictionary*.

---

#### 4.3.1 Modifying ACL Characteristics

You can use the DCL command SET ACL to create or modify an access control list (ACL) for a specified object. The format is as follows:

SET ACL/OBJECT\_TYPE=object

This command enables you to manipulate an entire access control list of an object, or to create, modify, or delete access control entries (ACEs) in the ACL of an object. The object name can specify a file (including a directory file) or a device.

The SET ACL command is used to add ACEs to an ACL by specifying the ACEs with the /ACL qualifier. For example, the following command adds an ACE to the ACL of the file SALARY86.DAT so that all users associated with the identifier PERSONNEL are allowed read access to the file:

```
$ SET ACL/ACL=(IDENTIFIER=[PERSONNEL],ACCESS=READ) SALARY86.DAT
```

If the object specified with the SET ACL command does not have an ACL, one is created.

The following example replaces the ACL of the file ACCOUNTS.LIS with the ACL for the file USER.LIS.

```
$ SET ACL/LIKE=(OBJECT_TYPE=FILE,OBJECT_NAME=USER.LIS) ACCOUNTS.LIS
```

As an alternative to the SET ACL command, you can use the DCL commands SET FILE/ACL and SET DIRECTORY/ACL to apply ACL protection to files and directories.

For a complete description of ACLs, see the *Guide to VAX/VMS System Security*. For a complete description of the Access Control List Editor, including information about the EDIT/ACL command and its qualifiers, see the *VAX/VMS Access Control List Editor Reference Manual*.

### 4.3.2 Modifying Directory Characteristics

You can use the DCL command SET DIRECTORY to modify the characteristics of one or more directories.

The examples that follow illustrate two instances of the SET DIRECTORY command:

---

#### EXAMPLES

**1** \$ SET DIRECTORY/VERSION\_LIMIT=5/CONFIRM [SMITH.FORTRAN]

The SET DIRECTORY command limits the number of versions to five for files created after the command is issued. The /CONFIRM qualifier requests that you confirm whether or not the specified directory should actually be modified.

**2** \$ SET DIRECTORY/OWNER\_UIC=[DOCUMENTATION,GRAY] [DAVIDSON],[USERS]

The SET DIRECTORY command modifies both the [DAVIDSON] and [USERS] directories, changing their owner UICs. Use of the /OWNER\_UIC qualifier requires SYSPRV or GRPPRV respectively for changing the ownership at the system or group level.

### 4.3.3 Modifying Disk File Characteristics

You can use the DCL command SET FILE to modify the characteristics of one or more files.

The examples that follow illustrate three instances of how the SET FILE command can be used to modify file characteristics.

---

#### EXAMPLES

1 \$ SET FILE/EXPIRATION\_DATE=15-APR-1986:11:00 BATCH.COM;3

The SET FILE command requests that the expiration date of the file BATCH.COM;3 be set to 11:00 A.M., April 15, 1986.

2 \$ SET FILE/BEFORE=15-APR-86/ERASE\_ON\_DELETE PERSONNEL\*.SAL

This SET FILE command calls for all files that match the file specification PERSONNEL\*.SAL and that are dated before April 15, 1986 to have their disk locations erased whenever one of them is deleted with commands such as DELETE or PURGE.

3 \$ SET FILE/OWNER\_UIC=[DOCUMENTATION,GRAY]/VERSION\_LIMIT=100 MYFILE.DAT

The SET FILE command modifies the characteristics of the file MYFILE.DAT, changing the owner UIC and assigning a file version limit of 100. Note that the /OWNER\_UIC qualifier requires SYSPRV or GRPPRV respectively for changing the ownership at the system or group level.

---

### 4.3.4 Modifying Magnetic Tape Device Characteristics

You can use the DCL command SET MAGTAPE to define the default characteristics associated with a specific magnetic tape device for subsequent file operations. The device must not be currently allocated to any other user.

The examples that follow illustrate three uses of the SET MAGTAPE command in conjunction with the MOUNT command.

---

#### EXAMPLES

1 \$ MOUNT MTB1:/FOREIGN  
\$ SET MAGTAPE MTB1: /DENSITY=800

The MOUNT command mounts a foreign tape on the device MTB1:. The SET MAGTAPE command defines the density for writing to the magnetic tape at 800 bpi. (The density is reset only if the magnetic tape has never been written before.)

- 2    \* MOUNT MTA0:/FOREIGN  
     \* SET MAGTAPE MTA0:/SKIP=FILES:4

The MOUNT command mounts a foreign magnetic tape on the device MTA0; the SET MAGTAPE command directs the magnetic tape position to skip four files.

- 3    \* MOUNT MTA1:/FOREIGN  
     \* SET MAGTAPE/REWIND MTA1:

The MOUNT command mounts a foreign tape on the device MTA1; the SET MAGTAPE rewinds the volume.

---

### 4.3.5 Modifying File Protection Characteristics

You can use the SET PROTECTION command to change or reset the protection characteristics for one or more files. If you include a protection code, the file protection is changed to that code. When you omit the protection code and do not use the /PROTECTION qualifier, the file protection changes to the default file access established by the SET PROTECTION/DEFAULT command. See the SET PROTECTION /DEFAULT command for information on how to change the default file protection.

All disk and magnetic tape volumes have protection codes that restrict access to the volume. The protection codes for magnetic tape volumes are assigned with the INITIALIZE and MOUNT commands. Protection characteristics on magnetic tape volumes cannot be changed by the SET PROTECTION command.

For disk volumes, each file on the volume can have a different protection associated with it. The SET PROTECTION command and other file-manipulating commands allow you to define the protection for individual files.

If you omit both the code and the /PROTECTION file qualifier, your current default protection (established by the SET PROTECTION/DEFAULT command) is applied to the file.

The examples that follow provide four instances of using the SET PROTECTION command.

## EXAMPLES

```
1 $ DELETE INCOME.DAT;3
  %DELETE-W-FILNOTDEL, error deleting DISK1:[SMITH]INCOME.DAT;3
  -RMS-E-PRV, insufficient privilege or file protection violation
  $ SET PROTECTION=OWNER:D INCOME.DAT;3
  $ DELETE INCOME.DAT;3
```

The file INCOME.DAT;3 has been protected against deletion. This SET PROTECTION command changes only the owner's DELETE access for the file INCOME.DAT;3. Now the file can be deleted.

```
2 $ SET PROTECTION -
  _$ PAYROLL.LIS/PROTECTION=(SYSTEM:R,OWNER:RWED,GROUP:RW),-
  _$ PAYROLL.OUT/PROTECTION=(SYSTEM:RWED,GROUP:RWED,W)
```

In this example, the SET PROTECTION command changes the protection codes applied to two files. To the file PAYROLL.LIS, it gives the SYSTEM READ access, the OWNER READ, WRITE, EXECUTE, and DELETE access, and users in the owner's group READ/WRITE access. To the file PAYROLL.OUT, it gives the SYSTEM and GROUP all types of access; the current access for OWNER does not change, but WORLD is denied all types of access.

```
3 $ SET PROTECTION A.DAT, B.DAT/PROTECTION=OWNER:RWED, C.DAT
```

The SET PROTECTION command specifies that the file A.DAT should receive the default protection established for your files. The existing protection for the file B.DAT is overridden, only for the OWNER category, to provide READ, WRITE, EXECUTE, and DELETE access. Note that no protection is specified for the file C.DAT at either the command or file level. Thus, like A.DAT, C.DAT receives the default protection.

Since no version numbers are specified in this example, the protection settings affect only the highest versions of the three files.

```
4 $ SET PROTECTION=OWNER:D -
  _$ [MALCOLM.SUB1]SUB2.DIR/PROTECTION=GROUP:D
```

The SET PROTECTION command changes the protection for the OWNER and GROUP categories of the subdirectory [MALCOLM.SUB1.SUB2] to permit deletion. However, the protection for WORLD and SYSTEM categories is not changed.

### 4.3.6 Modifying Volume Characteristics

You can use the DCL command SET VOLUME to modify the characteristics of one or more mounted Files-11 disk volumes. In order to use this command, you must have WRITE access to the index file on the volume. If you are not the owner of the volume, you must have either a system UIC or the user privilege SYSPRV. You must then specify the name of one or more mounted Files-11 volumes.

The examples that follow illustrate how the SET VOLUME command can be used.

#### EXAMPLES

1 \$ SET VOLUME/DATA\_CHECK=(READ,WRITE) DBC5

This command requests that data checks be performed following all read and write operations to DBC5:.

2 \$ SET VOLUME/LABEL=LICENSES DBC5:

This command encodes the label LICENSES on the volume DBC5:. Note that if characters in labels are entered in lowercase, they are changed to uppercase by the /LABEL qualifier.

### 4.4 Using DCL to Access Files

In addition to executing user programs that perform I/O operations to files residing on disk and magnetic tape volumes, you can use DCL commands to access disk and magnetic tape files for read and write operations.

Note that this section describes how to use DCL to access disk and magnetic tape files at the file level (as opposed to the record level). Although DCL does allow you to manipulate files at the record level, performance considerations usually warrant the use of a conventional programming language.

Section 4.5 contains a command procedure example that has been designed to access a volume in order to read and write records to files on the volume. This command procedure uses the DCL commands OPEN/READ and OPEN /WRITE to access individual records in a file.

Although you can use such a command procedure to manipulate individual records in a file, it is recommended that you write programs using the RMS facilities that are specifically designed to access files at the record level. You can write these programs in VAX MACRO or in any of the higher-level languages supported by VAX/VMS.

If you wish to access disk and magnetic tape files at the file level, you can take advantage of DCL. As mentioned above, you cannot use DCL commands to read or write files that are not in the standard formats supported by VAX/VMS. If the file formats are not standard, the volumes on which they reside must be mounted with the /FOREIGN qualifier.

The following sections provide examples of the steps you can use to access files on disk and magnetic tape volumes. In particular, these sections describe how to manipulate disk and magnetic tape files (at the file level) for READ and WRITE access.

---

### 4.4.1 Accessing Disk Files for Read and Write Operations

You can access disk files both for read and write operations. The next three sections show how you can use DCL to access disk files for both types of operations.

These sections contain examples of how to:

- Read files from a mounted disk volume
- Write files to a disk volume that must first be allocated, initialized, and mounted
- Write files from a default directory to a volume that must first be allocated, initialized, and mounted on a magnetic tape device.

Although the examples used below for accessing disk files focus on RK06/RK07 disk packs, they are applicable to other devices as well.

---

#### 4.4.1.1 Reading Files from a Disk Volume

In order to read the contents of a disk file, you can use the DCL command TYPE which displays the contents of a file on your terminal. To find the exact location of the disk file that you want to read, you can use the DCL command DIRECTORY, which was discussed in Section 4.2.1.

For example, if you wish to read the contents of a file named HISFILE, which is located somewhere in the directory [CHARLES] on a disk device whose logical name is DISK\$DOCUMENT:, you can look for the exact location of HISFILE by issuing the following command:

```
$ DIRECTORY DISK$DOCUMENT:[CHARLES...]HISFILE.*
```

This command instructs VAX/VMS to search the entire [CHARLES] directory, including all the subdirectories, for all file types and versions numbers of HISFILE. The following information will be displayed on your terminal.

```
Directory DISK$DOCUMENT: [CHARLES.MEMO]
HISFILE.UPD;1
Total of 1 file.
```

This display informs you that there is only one version of HISFILE, that its file type is UPD, and that this file resides in the [CHARLES.MEMO] directory.

To read the contents of this file, you can issue the following command.

```
* TYPE [CHARLES.MEMO]HISFILE.UPD
```

The contents of HISFILE will be displayed on your terminal.

---

### 4.4.1.2 Writing Files to a Disk Volume

Before you can write files to a disk volume, the volume must be properly prepared (see Chapter 3).

Disks are random-access devices, and files must be listed in directories. Therefore, after you initialize a disk, you must create a directory to contain your files on the disk volume.

For example:

```
* CREATE/DIRECTORY DMA3:[PUBS]
* DEFINE P DMA3:[PUBS]
* COPY *.* P
* COPY [PRIMER] *.* P
* COPY [COMMANDS] *.* P
```

The CREATE/DIRECTORY command creates a directory file named [PUBS] on the device DMA3:, and the DEFINE command defines the logical name P as DMA3:[PUBS]. The COPY command copies the highest versions of all files in the current default directory and in the directories [PRIMER] and [COMMANDS] to the newly created directory.

## 4.4.1.3 Writing Files from Disk Volumes to Magnetic Tape Volumes

The example below describes how to use DCL to write files from a default directory on a disk volume to an ANSI-labeled magnetic tape volume. It includes examples that show how to allocate, initialize, and use a magnetic tape to copy a set of your disk files. The procedures are similar to those outlined above for copying files from disk volumes to disk volumes. One main difference, however, is that magnetic tapes are sequential-access devices and do not have directories. (The characteristics of magnetic tape files are described more thoroughly in the next section.)

First, allocate a drive:

```
$ ALLOCATE MT: TAPE_DEVICE
%DCL-I-ALLOC _MARS$MTA2: allocated
```

This ALLOCATE command requests the allocation of a magnetic tape drive whose name begins with MT. TAPE\_DEVICE is a logical name, which in this case refers to MTA2:.

The system response indicates that unit 2 on controller A was available and is now allocated to you. You can now physically load the magnetic tape on the drive. Be sure the write ring on the magnetic tape is in place; if it is not, you cannot write to the magnetic tape.

Next, initialize the magnetic tape:

```
$ INITIALIZE TAPE_DEVICE: GMB001 -
_$ /PROTECTION=(GROUP:R,WORLD)
```

The INITIALIZE command specifies the device name (MTA2) and the volume label for the magnetic tape volume (GMB001). The label can be no longer than six characters. The /PROTECTION qualifier defines a protection code restricting group access to read and allowing no WORLD access. You can now use the MOUNT command to mount the volume and write files to it:

```
$ MOUNT TAPE_DEVICE: GMB001
%MOUNT-I-MOUNTED, GMB001 mounted on _MTA2:
$ COPY *.* TAPE_DEVICE:
```

The MOUNT command specifies the device name and volume label of the volume on the device. The COPY command copies the highest versions of all files in your default directory onto the magnetic tape. The file names, file types, and version numbers of the output files default to the same file names, file types, and version numbers as the input files.

If you issue the COPY command with the /LOG qualifier, the system will send a message to the current SYS\$OUTPUT device after each file has been copied. You can also use the DIRECTORY command to verify that the files were successfully copied.

```
$ DIRECTORY TAPE_DEVICE:
```

The DIRECTORY command lists the file names and file types of all files on the magnetic tape.

When you have finished using the magnetic tape, dismount and deallocate it as shown below:

```
$ DISMOUNT TAPE_DEVICE:  
$ DEALLOCATE TAPE_DEVICE:
```

If you do not dismount and deallocate the magnetic tape, the system does so automatically when you log out.

---

### 4.4.2 Accessing Magnetic Tape Files for Read and Write Operations

When you access an ANSI-labeled volume or a file, VAX/VMS checks at the volume and/or file level to ensure that your process can access the volume or file. The level at which VAX/VMS checks access depends on the operation that you request and the type of access that the operation requires.

When you access a volume or a file, VAX/VMS reads the volume and file header labels to determine whether access to the volume or file is restricted. Which label will be read depends on the operation that is requested. For example, if you want to mount a volume, your process must have access to it. Thus, VAX/VMS will read the volume labels only. For more detailed information on volume-level access (and protection requirements for magnetic tapes, see Chapter 2). This section describes file-level access for magnetic tapes.

The protection sets your access to a particular file. The expiration date field can prevent you from overwriting or appending to a file immediately preceding the one in question. If the date written in the expiration date field has not been reached, the file has not expired. You cannot overwrite an unexpired file unless you specify the /OVERRIDE=EXPIRATION qualifier when you mount the volume.

The manner in which the file is accessed to perform the operation is called the access type. READ and/or WRITE privilege is required, depending on the type of access required.

Before accessing a particular file for a read or write operation, you may want to search the magnetic tape volume for that file. The following section describes how you can use the DIRECTORY command to locate a file or group of files on a magnetic tape volume.

#### 4.4.2.1 Locating ANSI-Labeled Magnetic Tape Files for READ or WRITE Access

When you specify a VAX/VMS or ANSI file name for a file residing on magnetic tape, the magnetic tape file system compares the file name with the file header labels of each file until it finds a match in the file identifier field of the file header labels.

If you supply a version number in the file name, it is compared with the generation number and generation-version number fields in the first file header label. If you do not specify a version number, the magnetic tape file system neither defaults a version number nor checks the generation number and generation-version number fields. The magnetic tape file system selects the first file on the magnetic tape whose file name in the file identifier field matches the specified file name.

Neither the directory nor the latest version number concept is supported by VAX/VMS for magnetic tape volumes. VAX/VMS does not search for or list the latest version of a specified file. The magnetic tape file system cannot increment version numbers of files written to magnetic tape; therefore, two or more files in the same volume set can have the same file name and version number.

Because the magnetic tape file system selects the first matching file name and version number (if specified), the position of the magnetic tape within the volume set determines which file is returned on a search operation. A search operation begins at the current position, so you may want to rewind the volume set before accessing a file.

The search for a matching file and version number (if specified) continues at the beginning of the header label set of the next file. The search ends when the magnetic tape is positioned at the file where the search began. If the requested file is not found on the current volume, the remaining volumes in the volume set are searched sequentially, according to their relative volume numbers, until either a file name match occurs or the entire volume set is searched.

If a file name match occurs, the internal file identification number is constructed from the file section number, file sequence number, and relative volume number. Although you can access a disk file by its file identification number, you cannot access a magnetic tape file this way.

VAX/VMS does support the use of wildcard characters in file specifications for magnetic tape volumes. However, there are certain restrictions on using wildcard characters with magnetic tape files. Unlike files on disk volumes, which support the asterisk (\*), percent sign (%), ellipses (...), and minus sign (-) characters, magnetic tape volumes support only the asterisk and percent-sign wildcard characters with VAX/VMS file names. ANSI file names support only the asterisk wildcard character.

The asterisk wildcard character matches file specifications by field or portion of a field. The percent sign wildcard character matches any character in a file specification only by character positions within a field.

With VAX/VMS file names, you can specify the asterisk and the percent sign anywhere in the file name and file type field to match file name specifications by field or character position within a field. You cannot use the percent sign in the version number field. Only the asterisk wildcard character can be used in the version number field.

With ANSI file names, a single asterisk in a field is the only wildcard character that can be used. ANSI file names do support the special set of ASCII "a" characters. Unlike VAX/VMS file names, which can consist of up to 39 characters each for the file name and file type, ANSI file names can have a maximum of 17 characters in length. Whether you choose VAX/VMS file names or ANSI file names will depend on the type of applications you wish to perform.

The examples that follow provide three instances of how you can use wildcard characters in file specifications to search for files on magnetic tape volumes. These examples also show the DIRECTORY command can be used in conjunction with magnetic tapes. Note that the DIRECTORY command does not work the same with magnetic tape files as with disk files (see Section 4.2.1).

---

### EXAMPLES

1    \$ DIRECTORY MFA1:\*.;\*;

This command instructs VAX/VMS to search a volume set. Because asterisks are used in the file specification and the asterisk is a valid wildcard character for both ANSI and VAX/VMS file names, both VAX/VMS and ANSI file names will be returned. Note that ANSI file names will be returned within quotation mark characters.

2    \$ DIRECTORY MTA1:%\*.\*;\*;  
     \$ DIRECTORY MTAO:%\*.\*;\*;

In these two commands, the search can only match with VAX/VMS file names because the percent sign is not valid for ANSI file names. In the second command, the file type field must contain at least one character. Files with no file type are not returned.

3    \$ DIRECTORY MTAO:.\*.\*;

In this example, the DIRECTORY command instructs VAX/VMS to search for files with ANSI file names, as well as VAX/VMS file names that have a null file type.

#### 4.4.2.2 Reading Files on Magnetic Tape Volumes

When a magnetic tape file is accessed for a read operation, the magnetic tape is positioned at the beginning of the file section after the file header labels.

You can use the DCL command TYPE to read a file or group of files on the magnetic tape volume, and display the contents of the file on your terminal. For example, if you want to read the contents of a file named TESTFILE.DOC;1 (which you know from your above directory searches is a VAX/VMS file residing on the magnetic tape device MTA1:), you can enter the following command:

```
$ TYPE MTA1:TEST*.*.*
```

You will then receive the following display on your terminal:

```
MTA1:TESTFILE.DOC;1  
This is a test file.
```

When a file residing on a magnetic tape volume is accessed for reading only the attributes in the header labels, rather than the data in the file section, the magnetic tape file system returns the RMS attributes to your process. For example, when you specify the DIRECTORY/FULL command for a volume, file, or list of files, the magnetic tape file system selects the file identifiers from the header labels, returns the file attributes to your process, and positions the magnetic tape after the header labels of the last file accessed.

A magnetic tape file that was opened for read access is closed either implicitly or explicitly. The file is closed implicitly when the driver encounters a tape mark while a file is being read. The magnetic tape file system then reads the trailer labels, closes the file, and positions the magnetic tape at the next file.

The file is closed explicitly when you deaccess the file before all the data in the file is read. The magnetic tape file system will then close the file without reading the trailer labels, and the magnetic tape remains at the current position.

#### 4.4.2.3 Writing to Files on Magnetic Tape Volumes

When you use DCL to access an existing file for a write operation, one of the following operations is actually being performed: append or update. The difference between append and update write operations can be explained in the following way.

- **Append Access**

When a file is accessed for an append operation, the magnetic tape is positioned at the EOF before the tape mark that precedes the trailer labels. After the file is appended and closed, all files beyond the appended file are lost. When the positioning is complete, the processing is handled as if the file had been created as described in the section on file creation.

- **Update Access**

When a file is accessed for an update operation, the magnetic tape is positioned at the BOF section after the header labels. After the file is written to and closed, all files beyond the updated file are lost. The processing is handled as if the file had been created.

Note that you can only update/append magnetic tape files when the header label contains a value of zero for the buffer offset length. For more information on how to update/append magnetic tape files, see Chapter 5.

In addition to updating files and appending files on a magnetic tape volume, you can access a volume for a write operation by creating a file on the volume. You can use the CREATE command to write a new file to the magnetic tape volume. For example, you can enter the following command string:

```
$ CREATE MTAO:MYFILE
```

You can then write the contents of the file, without leaving the DCL command level, before closing the file.

If you do not specify the /OVERRIDE=EXPIRATION qualifier, the magnetic tape file system checks the expiration date field on the file before it allows you to write to that file. When more than one file will be overwritten, the magnetic tape file system also checks the expiration date of the file immediately following the file that will be overwritten. For example, before you append to a file, the magnetic tape file system checks the expiration dates of both the file being appended and the file immediately following. If the expiration date of either file has not been reached, the magnetic tape file system does not allow you to append the file.

When files are written to a magnetic tape volume, the magnetic tape file system performs access checks, writes labels, and, if necessary, switches volumes. If the new file will overwrite an existing file, the magnetic tape file system checks the expiration date and accessibility fields of the existing file. If overwriting is allowed, the magnetic tape file system overwrites the header label set of the existing file, creates the file section, writes the trailer labels, and writes two tape marks to denote the logical end-of-volume (EOV). All files following the newly created file are lost.

To close a magnetic tape file that was opened for WRITE access, the magnetic tape file system issues commands to the driver to write the labels, which are followed by a double tape mark that indicates the logical EOV.

---

### 4.5 Command Procedures for Accessing Foreign Volumes

The command procedures in this section allow you to create, read, or write magnetic tape data in a simple user-defined foreign format.

```
#!/
#!/      FOREIGN.COM
#!/
#!/ This is the master command procedure. It sets up the user account and
#!/ mounts the volume with the /FOREIGN qualifier. If the user wants to
#!/ read a foreign volume, the FORREAD.COM command procedure is called.
#!/ If the user wants to write a foreign volume the FORWRITE.COM
#!/ command procedure is called.
#!/
$      verify_off_on = F$VERIFY ( 0 )
$      ON CONTROL_Y THEN GOTO clean_up
$      ON WARNING THEN GOTO clean_up
$
#!/
#!/ If VOLPRO privilege is not set but the user account has SETPRV, VOLPRO is
#!/ set to allow the user to mount a new unformatted volume. If VOLPRO
#!/ privilege cannot be set, the user process is notified that the process
#!/ has insufficient privilege to write an unformatted volume. The user is also
#!/ asked whether to continue or exit the procedure.
#!/
$      volpro_off_on = F$SETPRV ( "VOLPRO" )
$      IF F$PRIVILEGE ( "VOLPRO" ) THEN GOTO cont
$      WRITE SYS$OUTPUT "Insufficient Privilege to write an unformatted volume!"
$      INQUIRE/NOPUNC continue "Do you wish to continue (Y|N) ? "
$      IF .NOT. continue THEN GOTO clean_up
$ cont:
```

## Manipulating Files on Disks and Magnetic Tapes

```

$
$!
$! Find out where the volume is mounted
$!
$ get_drive:
$     INQUIRE tape_drive "Tape drive"
$     IF tape_drive .EQS. "" THEN GOTO get_drive
$     tape_drive = tape_drive-"+"
$     IF .NOT. F$GETDVI (TAPE_DRIVE,"EXISTS")
$         THEN GOTO NOSUCHDEV
$     ASSIGN 'tape_drive' tape
$
$!
$! Try allocating and mounting the volume
$!
$     ALLOCATE tape:
$     MOUNT/NOASSIST/FOREIGN/OVERRIDE=(ACCESSIBILITY,EXPIRATION) tape:
$
$!
$! The user is asked whether a file will be read or written
$!
$ read_write:
$     INQUIRE/NOPUNC operation "Read or Write a file ? "
$     IF F$LOCATE ( operation, "READ" ) .EQ. 0 THEN GOTO read_op
$     IF F$LOCATE ( operation, "WRITE" ) .EQ. 0 THEN GOTO write_op
$     GOTO read_write
$
$ read_op:
$     @FORREAD
$     GOTO clean_up
$
$ write_op:
$     @FORWRITE
$
$ clean_up:
$
$!
$! Reset the user account the way it was before the command procedure began
$!
$     SET NOON
$     DISMOUNT/NOUNLOAD tape:
$     DEALLOCATE tape:
$     DEASSIGN tape
$     volpro_off_on = F$SETPRV ( volpro_off_on )
$     verify_off_on = F$VERIFY ( verify_off_on )
$
$ EXIT
$
$ NOSUCHDEV:
$     WRITE SYS$OUTPUT "No Such Device"
$     GOTO get_drive

```

## Manipulating Files on Disks and Magnetic Tapes

```

$! FORWRITE.COM    Writes Data to a Foreign Volume
$!
$! This command procedure writes data to a foreign volume.
$! The data format is as follows:
$!   Each record is a block.
$!   Records are variable length.
$!
$! The first four characters of a record are digits that are
$! padded on the left with spaces. This sequence field starts at
$! 1 and increases by 1 with each record.
$!
$! The fifth character of a record is a comma.
$!
$! The sixth through ninth characters are digits that are padded on the
$! left with spaces. This is a size field, which is the size of the data
$! field.
$!
$! The tenth character of a record is a vertical bar.
$!
$! The rest of the record is data.
$!
$! Records are padded to be at least 20 characters long.
$!
$
$   seq_num = 0      ! initialize the sequence number
$   spaces = "      " ! used to pad records less than 20 chars
$
$   OPEN/WRITE tape_file tape: ! open the output file
$ next_line:
$
$   seq_num = seq_num + 1 ! increment the sequence number
$
$! Prompt the user for the data
$!   close the file and exit if ^Z is entered
$!
$   READ/PROMPT="Record # 'seq_num' : "/END_OF_FILE=end_of_input -
$       SYS$COMMAND user_data
$
$! Find the size of the record and the number of pad characters needed
$! Note That a negative number of pad characters does not return any characters
$!
$   data_size = F$LENGTH ( user_data )
$   pad_chars = 10 - data_size
$   IF pad_chars .LT. 0 THEN pad_chars = 0
$
$! Construct the output record using FAO
$!
$   out_rec = F$FAO ( "!4UL,!4UL!AS", seq_num, data_size, user_data ) + --
$       F$EXTRACT ( 1, pad_chars, spaces )
$
$   WRITE tape_file out_rec ! write the formatted output record
$   GOTO next_line
$
$ end_of_input:
$   CLOSE tape_file

```

## Manipulating Files on Disks and Magnetic Tapes

```
#!/          FORREAD.COM    Reads Data from a Foreign Volume
#!/
#!/ This command procedure is called when data will be read. The procedure
#!/ reads data on a foreign volume. The data format is defined in
#!/ FORWRITE.COM command procedure.
#!/
$
$      seq_num = 0      ! initialize the sequence number
$
$      OPEN/READ tape_file tape: ! open the output file
$ next_line:
$
$      seq_num = seq_num + 1 ! increment the sequence number
$
$      READ/END_OF_FILE=end_of_input tape_file in_rec
$
$      record_num = F$EXTRACT ( 0, 4, in_rec ) -- " " -- " " -- " "
$      record_num = F$INTEGER ( record_num )
$      IF seq_num .NE. record_num THEN -
$          WRITE SYS$OUTPUT "Error possible data lost, record sequence broken!"
$
$!
$! Find the size of the data and extract the data
$!
$      data_size = F$EXTRACT ( 5, 4, in_rec ) -- " " -- " " -- " "
$      data_size = F$INTEGER ( data_size )
$      data_rec = F$EXTRACT ( 10, data_size, in_rec )
$
$      WRITE SYS$OUTPUT data_rec ! write the data output record
$      GOTO next_line
$
$ end_of_input:
$      CLOSE tape_file
```



# 5

## Transferring Disk and Magnetic Tape Information

This chapter describes the facilities that VAX/VMS provides for transferring information contained on disk and magnetic tape media. In particular, it describes how to use the DCL command COPY and the VAX/VMS Convert and Exchange Utilities to transfer disk and magnetic tape information. This chapter also contains examples of command procedures that are designed to transfer information on disks and magnetic tapes.

### 5.1 Transferring Information Within and Across Operating Systems

In many cases, you will be able to transfer information without physically transporting media. You may, however, find it necessary to transfer files between systems that are not connected by a communications link. In these circumstances, you must be able to physically move your files from one location to another. This is conveniently done by copying your files to a portable volume, such as a magnetic tape reel or disk pack, and then carrying that volume to the location of the other system.

The VAX/VMS operating system supports the transfer of information, contained on disks and magnetic tapes, both within the VAX/VMS system and across other operating systems. It provides a number of facilities to assist you in both types of information transfer. The two most frequently used facilities for transferring information are the DCL command COPY and the VAX/VMS Exchange Utility (EXCHANGE).

In many cases, you will find the COPY command and the Exchange Utility sufficient for accomplishing information transfers. You may also find the VAX/VMS Convert and Analyze/RMS\_File Utilities useful for transferring information, especially in foreign volume or non-file-structured environments.

Under some circumstances, you will need to use the Backup Utility (BACKUP) to transfer files. For example, BACKUP is the only means of transferring entire directory trees or files that are not sequentially structured using magnetic tape.

Each of the information transfer facilities provided by the VAX/VMS system, except BACKUP, is described in the sections that follow. This chapter also includes examples of command procedures that contain the commands and utilities provided by VAX/VMS for transferring information.

Chapter 6 provides information on the use of BACKUP, including examples of command procedures for performing backup tasks.

---

### 5.2 Using the COPY Command to Transfer Information

One way of transferring information on disk and/or magnetic tape media is to use the DCL command COPY. You can use the COPY command to copy files from:

- Disk to disk
- Disk to magnetic tape
- Magnetic tape to disk
- Magnetic tape to magnetic tape

The sections that follow describe how the COPY command is used with both disk and magnetic tape files.

---

#### 5.2.1 Copying Files from Disk Volumes

This section describes how to use the COPY command to copy files from disk volumes to other disk volumes and from disk volumes to magnetic tape volumes.

The default format for files on disk is called Files-11 Structure Level 2. You can also initialize disks in the Files-11 Structure Level 1 format. Structure Level 1 is the format used by other DIGITAL operating systems, including RSX-11M, RSX-11M+, RSX-11D, and IAS. The Files-11 structure for magnetic tapes is discussed in Section 5.2.2.

In the example that follows, the COPY commands copy the highest versions of all files in the current default directory and in the directories [PRIMER] and [COMMANDS] to the directory [PUBS].

```
$ DEFINE P DMA3:[PUBS]
$ COPY *.* P:
$ COPY [PRIMER]*.* P:
$ COPY [COMMANDS]*.* P:
```

If you wish to copy files from a disk directory such as your default directory on a public disk volume to a private Files-11 disk volume, you can also use the COPY command. Before copying these files, however, you must set up (allocate, initialize, and mount) a disk device as described in Chapter 3.

In the following example, assume that the disk device DMA5: has been allocated to your process and that a disk volume has been initialized and mounted on that device. Assume also that you have a directory called PRIVATE already created on that volume. You can copy all of the files in your default directory to the directory on that volume by entering the following command:

```
$ COPY *.* DMA5:[PRIVATE]
```

You can also use the COPY command to copy files from a disk volume to a magnetic tape volume. The procedure for copying files from disk volumes to magnetic tape volumes is similar to those outlined above for copying files across disk volumes. Note that magnetic tapes are sequential-access devices and do not have directories.

Again, you must set up a magnetic tape device (ALLOCATE, INITIALIZE, and MOUNT) before copying disk files to the magnetic tape volume. In the following example, assume that MTA2: has been allocated to your process and that a magnetic tape volume has been initialized and mounted on that device.

You can now use the COPY command to write files to the magnetic tape volume.

```
$ COPY *.* MTA2:
```

In this case, the highest versions of all files in your default disk directory are copied to the magnetic tape volume on MTA0:. The file names, file types, and version numbers of the output files default to the same file names, file types, and version numbers as the input files. Restrictions that apply to the use of version numbers for magnetic tapes are discussed in Section 5.2.2.

If you issue the COPY command with the /LOG qualifier, the system will send a message to the current SYS\$OUTPUT device after each file has been copied. To verify that the files were successfully copied, use the DIRECTORY command. For example, the following command will list the file names and file types of all files on the magnetic tape volume:

```
$ DIRECTORY MTA2:
```

---

### 5.2.2 Copying Files from Magnetic Tape Volumes

The default format for files on magnetic tapes is the ANSI-labeled volume. The VAX/VMS system supports sequential, relative, and indexed files on disks, but only sequential files can be copied to ANSI-labeled volumes. The only valid record formats are variable-length (ANSI D) and fixed-length (ANSI F).

Although VAX/VMS supports stream and VFC records, it encodes these records in a variable-length format on ANSI-labeled volumes. Non-VAX/VMS systems do not distinguish stream records from VFC records; instead, they interpret both as variable-length records. Therefore, neither stream nor VFC records should be created on volumes that will be used for information interchange to a non-VAX/VMS system.

When you copy files from disk to ANSI-labeled volumes, the following items are not preserved:

- Directory specifications
- Individual file protection
- User identification code (UIC)
- Creation time (but the date is preserved)
- Revision and backup dates and times

The full set of ASCII "a" characters are supported only for ANSI-labeled volumes, not for disk volumes. Therefore, when you copy files with ANSI file names from magnetic tape to disk, specify a standard VAX/VMS file name for the output file name specification.

If you do not specify a VAX/VMS file name on output, your process receives the following error message:

**RMS-F-FNM, error in file name**

This message indicates that the ANSI file name is not a valid file name specification.

The entire set of Files-11 file names is supported for magnetic tapes. You can copy a disk file with the following file name to a magnetic tape volume, without having to modify the file name:

**THIS\_IS\$A\_VAXVMSLONG\_FILE.LONG\_EXT**

The following examples illustrate ways of copying files to and from magnetic tape volumes.

---

**EXAMPLES**

1 \$ COPY/LOG MTA1:"%&\*?!SKI! "" SEASON.DAT  
 %COPY-S-COPIED, MTA1:[]"%&\*?!SKI! "".;1  
 copied to WRKD:[MANUAL]SEASON.DAT;1 (120 records)

Since the /LOG qualifier was specified, VAX/VMS returns a message that confirms the file was copied as specified and informs you how many records were copied. The ANSI file "%&\*?!SKI!#" (# means space) is copied to the file SEASON.DAT on the default disk and directory WRKD:[MANUAL]. The file could not have been copied to disk unless the new file name was specified. VAX/VMS provided defaults for segments of the file specification that were not specified.

2 \$ COPY/LOG FORTAP.DAT MTA1:"%&\*?!SKI! "" "  
 %COPY-S-COPIED, WRKD:[MANUAL]FORTAP.DAT;1  
 copied to MTA1:[]"%&\*?!SKI! "".;0 (120 records)

In this command an ANSI file name was specified as the output file specification. Note that VAX/VMS truncates the trailing space in the file name "%&\*?!SKI!##" (where # means space).

3 \$ COPY/LOG VAXVMS\_LONG\$FILE\_NAME.LONG\_EXT MTA1:  
 %COPY-S-COPIED, WRKD:[MANUAL]VAXVMS\_LONG\$FILE\_NAME\_EXT;1  
 copied to MTA1:VAXVMS\_LONG\$FILE\_NAME.LONG\_EXT;1 (80 records)

In this example, a VAX/VMS long file name with a long extension will be copied to the volume MTA1: with the same file name and type that had been on the disk volume.

4 \$ COPY %%.JOU;\* MTA1:.\*  
 %COPY-S-COPIED, WRKD:[MANUAL]C6.JOU;1 copied to MTA1:[]C6.JOU;1 (4 records)

In this example, all files with a 2-character file name and a file type of JOU will be copied to the volume MTA1: with the same file name and type as they had on the disk volume. Version numbers are preserved.

5 \$ COPY MTAO:.\* \*  
 %COPY-S-COPIED, MTAO:[]TASTETEST.DAT;1  
 copied to WRKD:[FOOD]TASTETEST.DAT;1 (249 records)  
 %COPY-S-COPIED, MTAO:[]CUKES.DAT;1 copied to WRKD:[FOOD]CUKES.DAT;1 (148 records)  
 %COPY-S-NEWFILES, 2 files created

In this example, neither file on the volume had an ANSI file name. Therefore, both files were copied to the disk volume.

```

6 $ COPY MTA1:*. * [SOURCE]
%COPY-S-COPIED, MTA1: []TAPETEST.DAT;1 copied to WRKD:[SOURCE]TAPETEST.DAT;1 (101 records)
%COPY-E-OPENOUT, error opening WRKD:[SOURCE]"%&*()*!SKI! """;1 as output
-RMS-F-FNM, error in file name
%COPY-W-NOTCOPIED, MTA1: []"%&*()*!SKI! """;1 not copied
%COPY-E-OPENOUT, error opening WRKD:[SOURCE]"SANFRAN%%""";1 as output
-RMS-F-FNM, error in file name
%COPY-W-NOTCOPIED, MTA1: []"SANFRAN%%""";1 not copied
%COPY-S-COPIED, MTA1: []VAXVMS_LONG$FILE_NAME.LONG_EXT;1
copied to WRKD$: [SOURCE]VAXVMS_LONG$FILE_NAME.LONG_EXT;1 (80 records)
%COPY-S-COPIED, MTA1: []C6.JOU;1 copied to WRKD:[SOURCE]C6.JOU;1 (4 records)
%COPY-S-NEWFILES, 2 files created

```

The COPY command string specifies that all files on the volume mounted on drive MTA1: should be copied to the current default disk and directory WRKD:[SOURCE]. However, files with ANSI file names are not copied; VAX/VMS returns an error message to the process.

The following sections provide guidelines for interchanging volumes.

### 5.2.3 Copying Files to and from Non-File-Structured Volumes

The VAX/VMS operating system supports the transfer of files between file-structured and non-file-structured volumes. You can use the COPY command to transfer files in the following ways:

- from file-structured volumes to non-file-structured volumes
- from non-file-structured volumes to file-structured volumes

The next two sections contain examples of each type of transfer.

#### 5.2.3.1 Copying Files to a Non-File-Structured Volume

This section contains a procedure for copying files from a file-structured volume to a non-file-structured or foreign volume.

The procedure copies three files from an ANSI-labeled volume to a foreign volume.

- 1 Mount the volumes involved in the copy operation.

```

$ MOUNT MTA1: FRESKI
% MOUNT-I-MOUNTED, FRESKI mounted on _MTA1:
$ MOUNT/FOREIGN MTAO:
%MOUNT-I-MOUNTED, mounted on _MTAO:

```

- 2 Specify the COPY command string, including the full device and file name specifications in the input specification, but specify the device name only in the output specification. File names are specified on input because the volume from which files will be copied is ANSI-labeled and file structured. However, only the device name is specified on output because the volume to which the files will be copied is foreign and is not file structured.

```
$ COPY/LOG MTA1:PROG1.EXE, PROG2.EXE, PROG3.EXE MTA0:  
%COPY-S-COPIED, MTA1:[]PROG1.EXE;1 copied to MTA0: (92 records)  
%COPY-S-COPIED, MTA1:[]PROG2.EXE;6 copied to MTA0: (70 records)  
%COPY-S-COPIED, MTA1:[]PROG3.EXE;2 copied to MTA0: (77 records)  
%COPY-S-NEWFILES, 3 files created
```

Because no version numbers were specified for the files copied from the ANSI-labeled magnetic tape volume, the first version of each specified file that the MTAACP finds on the ANSI-labeled volume will be the version of the file that is copied to the foreign volume.

When you use the /LOG qualifier with the COPY command, the system informs you which files are copied and tallies the number of records copied and the number of files created.

When you copy files as shown in this example, the structure of the resulting magnetic tape is referred to as unblocked, non-file-structured. Each file record becomes a physical record on the magnetic tape. Each file boundary is indicated on the magnetic tape by a single tape mark. Two consecutive tape marks indicate end of volume. No label, file name, or attribute information is present.

### 5.2.3.2 Copying Files from a Non-File-Structured Volume

This section contains a procedure for copying data segments from a non-file-structured or foreign volume to a file-structured ANSI-labeled volume.

Note that you must specify a COPY command for each segment of data that you copy from a foreign or non-file-structured volume to a file-structured volume.

- 1 Mount the volumes as follows:

```
$ MOUNT/NOLABEL MTA1: " " COLD:  
%MOUNT-I-MOUNTED, mounted on _MTA1:  
$ MOUNT MTA0: FEVER SEASON:  
%MOUNT-I-MOUNTED, FEVER mounted on _MTA0:
```

- 2 Specify a COPY command for each segment on the foreign volume that will be copied to files on the file structured volume. Because the volume to which data segments are copied is file structured, you must specify a file name in the output file specification.

```
$ COPY/LOG COLD: SEASON:FILE1.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE1.DAT;1 (92 records)
$ COPY/LOG COLD SEASON:FILE2.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE2.DAT;1 (62 records)
$ COPY/LOG COLD: SEASON:FILE3.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE3.DAT;1 (23 records)
$ COPY/LOG COLD: SEASON:FILE4.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE4.DAT;1 (48 records)
$ COPY/LOG COLD: SEASON:FILE5.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE5.DAT;1 (37 records)
$ COPY/LOG COLD: SEASON:FILE6.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE6.DAT;1 (10 records)
```

Since the /LOG qualifier was used with the COPY command, VAX/VMS informs you that the files you specified were copied to the appropriate device, and tallies the number of records transferred.

---

### 5.3 Using the Convert Utility to Transfer Information

This section describes how to use the VAX/VMS Convert Utility (CONVERT) to transfer information to non-file-structured or foreign volumes. It also describes how the VAX/VMS Analyze/RMS\_File Utility (ANALYZE/RMS\_FILE), which is useful in determining record size, is used in conjunction with CONVERT.

There may be cases in which you will be unable to transfer information to a non-file-structured or foreign volume by using the COPY command. For example, files containing records smaller than 14 bytes cannot be copied successfully to a foreign volume with the COPY command. However, they can be transferred by using CONVERT.

CONVERT allows you to pad and extend the record size to make it suitable for a foreign volume. The following procedure illustrates how to use the Convert Utility.

#### Procedure

- 1 Mount both volumes. Specify the /RECORDSIZE and /BLOCKSIZE qualifiers in addition to either the /NOLABEL or /FOREIGN qualifier when you mount the foreign volume.

```
$ MOUNT/FOREIGN/RECORDSIZE=80/BLOCKSIZE=2400 MTA1:
%MOUNT-I-MOUNTED, mounted on _MTA1:
$ MOUNT MTA0: LATER TODAY
%MOUNT-I-MOUNTED, LATER mounted on _MTA0:
```

VAX/VMS informs you that the foreign volume is mounted on drive MTA1:, and that the target volume is mounted on MTA0:. The /RECORDSIZE and /BLOCKSIZE specify that tape will be written as 80-byte records that are blocked in 2400-byte blocks.

- 2 Check the size of the largest record within a file. To do so, use the Analyze/RMS\_File Utility as shown in the following example:

```
$ ANALYZE/RMS_FILE/STATISTICS TODAY:TICKET.DAT
RMS File Statistics                      14-JUN-1985 21:33:20.07
TODAY: []TICKET.DAT;3

FILE HEADER
  File Spec: TODAY: []TICKET.DAT;3
  File ID: (1,1,1)
  Owner UIC: [012,141]
  Protection: System: RWED, Owner: RWED, Group: RWED, World: RWED
  Creation Date: 5-APR-1985 00:00:00.00
  Revision Date: 17-NOV-1858 00:00:00.00, Number: 0
  Expiration Date: none specified
  Backup Date: none posted
  Contiguity Options: none
  Performance Options: none
  Reliability Options: none

RMS FILE ATTRIBUTES
  File Organization: sequential
  Record Format: variable
  Record Attributes: carriage-return
  Maximum Record Size: 0
  Longest Record: 77
  Blocks Allocated: 0, Default Extend Size: 0
  End-of-File VBN: 1, Offset: %X'0000'
```

The analysis uncovered NO errors.

```
ANALYZE/RMS_FILE/STATISTICS TODAY:TICKET.DAT
```

- 3 Extend the size of records smaller than 14 bytes to that specified in the MOUNT command and ensure the record size that you specify is sufficient. To do so, specify the following commands:

```
$ CONVERT/PAD/FDL=SYS$INPUT
$_Input: TODAY:TICKET.DAT
$_Output: MTA1:
RECORD
SIZE 80
FORMAT FIX 15 EXIT
$
```

Specifying the above CONVERT command string and input and output parameters pads and extends the size of records when copying them from the ANSI-labeled volume mounted on MTA0: to the foreign volume mounted on MTA1:. CONVERT command and qualifiers can pad short records to correspond to the record size specified by the

MOUNT command in the first step. After you specify the command line, CONVERT prompts you for input and output parameters. At the `$_Input:` prompt, supply the specification of the file containing the records that were too short. At the `$_Output:` prompt, specify only the physical or logical device name of the foreign volume. Specifying CTRL/Z terminates the file description and causes CONVERT to run. To verify that the file was copied to the foreign volume, you can rewind the foreign volume and type the volume as shown in the steps below.

- 4 To rewind a foreign volume, specify the SET MAGTAPE/REWIND command string shown below.

```
$ SET MAGTAPE/REWIND MTA1:
$
```

- 5 To verify the contents of a file, specify the TYPE command as shown below.

```
$ TYPE MTA1:
This is a test file.
Some records contain less than 14 bytes.
This file was copied to a foreign volume with the
RMS Convert Utility.
Any record less than 80 bytes was extended and padded to
80 bytes.
```

The TYPE command types the first data segment on the volume. If you copy multiple files to a foreign volume with the RMS Convert Utility, each file is copied to the volume in a separate data segment. Therefore, you must specify a TYPE command for each data segment copied to the volume until you reach the segment that you want to verify.

This example produces what is referred to as a blocked non-file-structured volume. Each record of the input file is placed into an 80-byte area of a 2400-byte tape block. Tape marks are used to delimit files, as in the previous example.

---

### 5.4 Using the Exchange Utility to Transfer Information

The VAX/VMS system provides a utility called Exchange, which enables you to read to or write from disk and magnetic tape media being interchanged with DIGITAL-supported, non-VAX/VMS operating systems. With the VAX/VMS Exchange Utility (EXCHANGE), you can manipulate mass-storage volumes that are written in formats other than those normally recognized by VAX/VMS. You can also use EXCHANGE to manipulate Files-11 files that are images of foreign volumes.

You can use EXCHANGE to transfer files between foreign volumes and VAX/VMS file-structured volumes. You can also use EXCHANGE to perform volume-specific initialization and manipulation functions on the foreign volumes. The Exchange Utility enables you to convert the format of the files, as appropriate, when transferring files between volumes with different structures.

You can use the Exchange Utility to perform file transfers and format conversions for:

- DOS-11 magnetic tape volumes
- Files-11 volumes
- RT-11 block-addressable volumes

In addition to transferring files, the Exchange Utility allows you to mount and dismount foreign volumes, initialize foreign volumes, list directories of volumes, delete files from block-addressable volumes, and locate bad blocks on the volume. For further information on how to use EXCHANGE to accomplish these tasks, see the *VAX/VMS Exchange Utility Reference Manual*.

### 5.4.1 Invoking and Terminating the Exchange Utility

To invoke EXCHANGE, enter the following command in response to the DCL prompt:

```
$ EXCHANGE  
EXCHANGE>
```

When you receive the EXCHANGE prompt (EXCHANGE> ), you are in the Exchange Utility. Once you are in this utility, you can enter any of the command strings supported for EXCHANGE. For example:

```
EXCHANGE> DIRECTORY MFA0:/VOLUME=DOS11/FULL
```

This command lists all the files on the DOS-11 magnetic tape mounted on MFA0:. In this case, the magnetic tape will be rewound before the files are listed.

The following example illustrates the use of the MOUNT command within the EXCHANGE utility:

```
EXCHANGE> MOUNT DMA1:  
%EXCHANGE-I-WRITELOCK, volume is write-locked  
%EXCHANGE-I-MOUNTED, volume DMA1: mounted
```

This command mounts the foreign volume that is loaded in the RK07 device DMA1:, making the volume available for subsequent commands. In this case, EXCHANGE recognizes that the volume itself is write-locked, and displays the message.

For a complete list of all the commands, qualifiers, and parameters that are supported for EXCHANGE, see the *VAX/VMS Exchange Utility Reference Manual*.

To exit from EXCHANGE and return to DCL level, you can use the EXCHANGE command EXIT or CTRL/Z.

---

### 5.4.2 Using EXCHANGE at DCL Command Level

In addition to using EXCHANGE interactively as a utility, you can use EXCHANGE as a DCL command. To do this, you append a command string to the DCL command EXCHANGE as follows:

```
$ EXCHANGE DIRECTORY DMA1:/VOLUME_FORMAT=RT11
```

This DCL command lists the directory of the RT-11 volume mounted (/FOREIGN) on DMA1:. At DCL level, EXCHANGE executes the single command and returns to the DCL prompt.

At DCL level, you can process only one command at a time. Some tasks for which EXCHANGE was designed, however, require more than one EXCHANGE command. For example, the use of virtual devices requires multiple EXCHANGE commands.

Tasks requiring more than one EXCHANGE command cannot be performed at DCL level. You can, however, design a command procedure to execute more than one EXCHANGE command for a particular task. In that case, you could execute the command procedure without leaving the DCL command level. For further details on how to construct command procedures using EXCHANGE, see Section 5.5.2.

---

## 5.5 Using Command Procedures to Transfer Information

You can design command procedures to facilitate routine transferring of information on disk and magnetic tape media.

Two examples of command procedures are included in this section. The first example is designed to transfer information by way of the COPY facility. The second example employs the Exchange Utility to transfer the information.

### 5.5.1 Using a Command Procedure to Copy Files

The sample command procedure in this section can be used for a variety of purposes. This procedure can copy from disk to similar disk, from disk to dissimilar disk, and from magnetic tape to magnetic tape.

Note that the command procedure mounts the volume as non-file-structured.

If you are using magnetic tape media, this command procedure will not work for multi-reel magnetic tape volume sets. Magnetic tape to magnetic tape transfer is supported for single volumes only.

The magnetic tape to magnetic tape copy is done using the asterisk (\*) wildcard character.

```
$ vol_priv = f$setprv("volpro")
$ Type sys$input
$ on control_y then goto cleanup
$ on warning then goto cleanup
$
$ get_source:
$   write sys$output "Enter drive holding the write-locked master."
$   inquire source "[CS1$DRB0:, DBxx:, DDxx:, DJxx:, DLxx:, DMxx:, DQxx:, DRxx:, DUxx:, DYxx:,
$     MFxx:, MTxx:, MUxx:]"
$   if source .eqs. "" then goto get_source
$   source = source -": "+"":
$
$ get_target:
$   inquire target "Enter drive holding the write-enabled target."
$   if target .eqs. "" then goto get_target
$   target = target -": "+"":
$   kit_type := 'f$extract(0,1,source)
$   if f$locate("CS",source) .eq. f$length(source) .and. -
$     f$locate("CS",target) .eq. f$length(target) then goto prepare
$
$ ! *Note* -- This section checks to see if the console is configured into the system.
$ !           If it does not exist, SYSGEN will be run to configure it in
$
$ connect_console:
$   ! (NOTE* For 750,730 media copying CONSOLE must be configured)
$   if f$getdvi("CS:1","exists") then goto prepare
$   set noon
$   proc_priv = f$setprv("cmkrnl,cmexec")
$   run sys$system:sysgen
$   connect console
$   proc_priv = f$setprv(proc_priv)
$   set on
```

## Transferring Disk and Magnetic Tape Information

```
$
$ prepare:
$   allocate 'source'
$   inquire start "Ready source volume ''source' for mounting and press return"
$
$ begin:
$   allocate 'target'
$   mount/foreign 'source'
$   inquire start "Ready target volume ''target' for initialization and press return"
$   volume_lbl = f$getdvi(source,"VOLNAM")
$   if (volume_lbl .eqs. "") then volume_lbl := "console"
$   init 'target' 'volume_lbl'
$   If (kit_type .eqs. "M") then goto tape_copy
$   mount/foreign 'target'
$   backup/physical/verify 'source' 'target'
$   goto copy_done
$
$ tape_copy:
$   mount/over:id 'target'
$   copy 'source'*. * 'target'*/log
$
$ copy_done:
$   dismount 'target'
$   deallocate 'target'
$   write sys$output "Copy is complete"
$   dismount/nounload 'source'
$   inquire answer "Do you want to make another copy of the Source volume?"
$   if .not. answer then goto new_source
$   write sys$output "Please place volume in the target device ''target'." "
$   goto begin
$
$ new_source:
$   deallocate 'source'
$   inquire answer "Do you want to make a copy using a new Source device?"
$   if answer then goto get_source
$   goto finish
$
$ cleanup:
$   if f$getdvi(target,"mnt") then dismount 'target'
$   if f$getdvi(source,"mnt") then dismount 'source'
$
$ finish:
$   vol_priv = f$setprv(vol_priv)
$   if f$getdvi(target,"all") then deallocate 'target'
$   if f$getdvi(source,"all") then deallocate 'source'
$   exit
```

### 5.5.2 Using a Command Procedure to Exchange Information

This command procedure is designed to exchange files between the console device and the current directory on disk.

The files that are to be copied are assumed to be in standard format as determined by file type.

```
$ WRITE SYS$OUTPUT " Command file to copy files to/from the system"
$ WRITE SYS$OUTPUT " console storage medium and the current directory."
$ WRITE SYS$OUTPUT " "
$ INQUIRE MOUNT "Is system console storage medium mounted (Y/N)?"
$ IF MOUNT THEN GOTO MOUNTED
$ WRITE SYS$OUTPUT "Please place the system console medium in the console drive"
$ INQUIRE MOUNT "and type <RET> when ready"
$ RUN SYS$SYSTEM:SYSGEN
CONNECT CONSOLE
$ MOUNT/SYSTEM/FOREIGN CSA1: "VAX console"
$
$ MOUNTED:
$ INQUIRE DIR "Copy from console medium (Y/N)?"
$ IF DIR THEN GOTO FROMCON
$ INQUIRE SOURCE "Enter file name(s)"
$ IF SOURCE .EQS. "" THEN GOTO EXIT
$ EXCHANGE COPY /LOG 'SOURCE' CSA1:
$ GOTO EXIT
$
$ FROMCON:
$ INQUIRE SOURCE "Enter console file name"
$ IF SOURCE .EQS. "" THEN GOTO EXIT
$ EXCHANGE COPY /LOG CSA1:'SOURCE' *
$ EXIT:
$ DISMOUNT CSA1:
$ MOUNT/SYSTEM/FOREIGN/NOWRITE CSA1: "VAX console"
```



# 6

## Maintaining Data Integrity on Disk and Magnetic Tape Volumes

This chapter discusses methods of maintaining data integrity on disk and magnetic tape media. In particular, it focuses on how to use the VAX/VMS Backup Utility (BACKUP) to perform routine types of data integrity operations.

The term *integrity*, as it applies to disk and magnetic tape media, means unimpaired condition of the data on the volumes. Maintaining that integrity means ensuring that the data contained in those volumes is protected against both accidental and deliberate corruption.

You will find that it is good practice to employ a number of methods to ensure the safety of your files and volumes. For example, you can set the type of protection you need on individual files and volumes, and you can establish user identification codes (UICs). By setting protection or employing UICs, you control the extent of privileges given to others who use your files or volumes. Protection codes and security-related topics are discussed in the *Guide to VAX/VMS System Security* as well as in Chapter 2 of this guide.

One of the most effective ways you can prevent loss is by creating functionally equivalent copies of the originals — that is, by “backing up” your files and volumes. You can do this by using the VAX/VMS Backup Utility. If you use BACKUP on a regular basis, you reduce the chances of losing data that might otherwise be destroyed by volume or file corruption. You should store the backup copies in a secure place so that if all else fails, you still have an uncorrupted version of the original.

Protection is not the only reason for backing up files, however. If you have limited disk space, it is often practical to store seldom used disk files on magnetic tape until you need to retrieve them.

The following section provides an introduction to the VAX/VMS Backup Utility and the various types of backup media. Subsequent sections describe how to perform backup tasks that are frequently used. For detailed information on BACKUP, including descriptions of command qualifiers, parameters, and restrictions, refer to the *VAX/VMS Backup Utility Reference Manual*.

This chapter does not include standalone BACKUP, which is used primarily as a system management tool. For information on using standalone BACKUP, refer to the *VAX/VMS System Manager's Reference Manual*.

## 6.1 Using BACKUP to Maintain Data Integrity

In order to keep your backup copies up to date, you should perform backups on a regular basis. The VAX/VMS Backup Utility is designed to help you with the process of creating and retrieving backup copies. Depending on both the nature of the data on your volumes and the frequency of change to the data, you can choose to back up your files as frequently or infrequently as you like. You can accomplish this through the use of various BACKUP command qualifiers.

The operation you choose depends on the nature of your work or the sensitivity of the data in your files. You also have the option of performing selective backup operations through the use of a single DCL command. With selective backup operations, you tell BACKUP to select only the files that meet specific criteria, such as version number, file type, UIC, date and time of creation, expiration date, or modification date. You can save a great deal of time and storage space by performing selective backups on a regular basis, rather than backing up entire volumes on which only a small percentage of files have been modified.

Backup operations that select only those files that have been modified since the last backup operation are called incremental backups. In a typical operating environment, a system manager or operator might perform incremental backups either daily or two or three times a week, then perform full backups (the entire volume) at the end of each week. For detailed information on performing incremental backups, see the *VAX/VMS System Manager's Reference Manual*.

You can use BACKUP to perform five basic types of operations: COPY, SAVE, RESTORE, COMPARE, and LIST. You will find it helpful to gain an understanding of how these operations differ from each other in order to determine your particular backup needs (Section 6.2.2 provides a description of each). You have the option of using some of these operations together in one command line. For example, you can use the list operation in conjunction with the copy operation to provide a list of the files as they are being copied.

You will also find it helpful to understand the use of BACKUP media (Section 6.3 provides a description of BACKUP media and save sets). For example, you can choose to back up your data as a functionally equivalent copy on either a Files-11 structured disk in standard format; or as a save set on magnetic tape, a standard Files-11 structured disk, or a sequential disk. It is especially important to recognize the difference between a functionally equivalent copy and a backup save set.

The remainder of this chapter describes the kinds of media that you can use with BACKUP, the various types of backup operations that are frequently used in a VAX/VMS environment, and the steps you need to take to perform these operations.

---

## 6.2 Choosing Backup Operations

When you perform backups, you have two basic decisions to make regarding the operation you wish to perform:

- 1 *What* you want to back up
- 2 *How* you want BACKUP to perform the operation

The following sections discuss the choices.

---

### 6.2.1 Deciding *What* to Back Up

You have the option of performing backups in the following modes of operation:

- **File by file**  
You can back up individual files or directories.
- **Image**  
An image operation allows you to create a functionally equivalent copy of the entire volume or volume set.  
The image backup requires more time to perform than a physical backup (see the description of physical backups below), but an image backup offers two significant features that are not available with a physical backup:
  - 1 When an image copy operation is performed (or an image save set is restored), the files and free space are reorganized and compacted.
  - 2 You can selectively restore individual files.
- **Physical**  
Like the image operation, the physical backup operation copies, saves, restores, or compares the entire volume. However, instead of creating a functionally equivalent duplicate, it creates an exact duplicate by ignoring the file structure on the disk and copying logical blocks.  
A physical backup is considerably faster than an image backup in most cases, but an image backup offers significant features that are unavailable with a physical backup (see the description of image operations above).

- **Selective**  
BACKUP can process files or volumes selectively, according to criteria such as version number, file type, UIC, date and time of creation, expiration date, or modification date.
- **Incremental**  
The incremental backup saves only those files that were created or modified since the last backup. Incremental backups are typically performed by system managers or operators on a regular basis. For detailed information on performing incremental backups, see the *VAX/VMS System Manager's Reference Manual*.

---

### 6.2.2 Deciding *How* to Back Up Files or Volumes

Once you know what you want to back up, you need to determine how you want BACKUP to operate — that is, what type of BACKUP operation is best suited to your needs. The five choices of operation are as follows:

- Copy
- Save
- Restore
- List
- Compare

You define the type of operation you want to perform when you enter the DCL command string. The input and output specifiers and the qualifiers you use indicate to BACKUP what type of operation to perform.

The remainder of this section describes the various operations performed by the Backup Utility.

---

#### 6.2.2.1 Copying Files

The copy operation allows you to create an equivalent copy of files, directories, directory structures, and disks.

You should be aware that the BACKUP copy operation produces results that are different from those of the DCL command COPY. The DCL command COPY makes new copies of files, updating the revision dates and assigning protection from the applicable defaults. The backup copy is, however, identical in all respects to the original, including date and protection.

---

#### 6.2.2.2 Saving Files

A BACKUP save operation places a selected disk file or directory into a save set on disk or magnetic tape. A save set is a file created by the VAX/VMS Backup Utility and written in a format that only BACKUP can interpret. You will find more information on save sets in Section 6.3, which discusses backup media.

---

#### 6.2.2.3 Restoring Files

The BACKUP restore operation enables you to recover the data that resides in a BACKUP save set. The restore operation converts the data back to its original format and sends the file to the output device that you specify.

---

#### 6.2.2.4 Comparing Files

After you have performed a BACKUP copy, save, or restore operation, you may want to check the integrity of the file or volume you have processed. One way to do this is by using the BACKUP compare operation, which can compare a save set with files on disk, or files on disk with other files on disk. For example, after a save operation you can compare the save set you have created against the original files.

You should be aware that the compare operation is intended for use only with files or volumes that were copied using BACKUP. Comparing sets of files that were not produced by BACKUP can cause errors, even though the files may otherwise appear to be identical.

---

#### 6.2.2.5 Listing Files

If you need information concerning the contents of a save set, you can use the BACKUP list operation. Because save sets are written in a format that can be interpreted only by BACKUP, the list operation is the only way you can determine the contents of a save set without actually restoring it first. The information, which can be sent to either your terminal or a specified output file, includes the date and time the save set was created, the name of the person who created it, and the names of the files it contains.

In addition, you can perform the list operation in conjunction with any other BACKUP operation.

---

### 6.2.3 Invoking the VAX/VMS Backup Utility

All of the BACKUP commands are issued at the DCL command level (except for standalone BACKUP). For most operations, the DCL command BACKUP requires an input specifier and an output specifier.

Generally, BACKUP assumes that an input or output specifier that refers to a Files-11 disk is a standard VAX/VMS file specification; the /SAVE\_SET qualifier is used to specify a save set on a Files-11 volume. An input or output specifier that refers to magnetic tape is always a save set. You cannot use save-set specifiers for both the input specifier and output specifier. Save-set specifiers can include node names; file specifications cannot.

The input and output specifiers given in the command line indicate the type of backup operation to be performed. In addition, various qualifiers are available: BACKUP command qualifiers, input specifier qualifiers, and output specifier qualifiers (see the *VAX/VMS Backup Utility Reference Manual*).

---

## 6.3 Using BACKUP Media

The number of parameters you include with the BACKUP command depends on the type of operation you are performing. For a copy, save, restore, or compare operation, you need an input specifier and an output specifier; a list operation requires only an input specifier. These specifiers refer to the media (disk or magnetic tape) on which you are performing the backup operation. If your input or output specifier refers to a disk, the data may be in either standard Files-11 disk structure format or in the format of a backup save set; if the specifier refers to a magnetic tape or a sequential disk, the data must be in the format of a save set.

A save set specifier is a file specification that identifies a file containing data in BACKUP format; there is no default file type for save-set specifiers.

You can write save sets to the following types of media:

- Magnetic tape
- Files-11 disk
- Sequential disk
- Remote Files-11 disk

The following sections discuss how you can use BACKUP with magnetic tape, Files-11 disks, and sequential disks.

---

### 6.3.1 Using BACKUP with Magnetic Tape

Magnetic tape is the most commonly used medium for storing backup data. It is less expensive than disk media, and its compact size makes it practical for storage purposes.

You can use more than one tape device at a time to save or restore data; this allows processing to continue on another tape while the one most recently used is rewinding. If no save-set name is given for an input magnetic tape, the first save set encountered on the tape will be read.

Before you write save sets on magnetic tape, you must remember to mount the magnetic tape volume as a foreign device. You do this by using the DCL command MOUNT/FOREIGN (more information on the VAX/VMS Mount Utility is available in the *VAX/VMS Mount Utility Reference Manual*). Magnetic tape volumes generally do not need to be initialized; however, if you plan to use BACKUP with a fresh magnetic tape, you will need to first initialize the magnetic tape by including the /REWIND qualifier with the BACKUP command. As an alternative, you can use the DCL command INITIALIZE before mounting the magnetic tape. For more information concerning magnetic tape volumes, see Section 6.4.2.

Save-set names on magnetic tape are limited to 17 characters, including the delimiter (.) and file type. For example:

27SEP1985SAV.WORK

The save-set file name consists of 12 characters; therefore, the file type is limited to 4 characters.

---

### 6.3.2 Using BACKUP with Files-11 Disk Volumes

You can write save sets on a Files-11 disk, using standard Files-11 format. If you are writing to a disk volume set, all volumes in the set must be mounted, as described in Chapter 3.

You indicate to BACKUP that you want to create a save set by using a qualifier, /SAVE\_SET, in the DCL command string. If you do not use the /SAVE\_SET qualifier, BACKUP assumes that you want to create a copy on the output volume, rather than a save set of the selected files.

### 6.3.3 Using BACKUP with Sequential Disk Volumes

Sequential disk save sets allow you to treat a Files-11 disk volume as you would treat a magnetic tape save set. The primary advantage of using sequential disk save sets is that you can mount multivolume save sets one volume at a time; this is particularly useful on systems without tape that have a large fixed-media disk and a small removable disk. When one sequential disk is full, BACKUP prompts you for another disk, as it would for save sets on magnetic tape.

To write sequential disk save sets, you must mount the output disk using the DCL command MOUNT/FOREIGN. Although the disk is mounted with the /FOREIGN qualifier, BACKUP manages the disk using Files-11 structure. When you perform a save operation to a disk, you must use the /SAVE\_SET output qualifier. You must not specify a directory name for the save set; save sets are entered in the master file directory (MFD) on the disk. You need the user privilege LOG\_IO to read or write a multivolume sequential disk save set. You can use more than one disk device at a time to save or restore data; this allows processing to continue on another disk while the one most recently used is spinning down.

BACKUP does not initialize the first sequential disk volume, since the default is /NOINITIALIZE; however, continuation volumes are always initialized. Unless you use the /INITIALIZE qualifier, the following restrictions apply to the output volume:

- The disk must be Files-11 Structure Level 2.
- The disk must not be part of a normal Files-11 volume set.
- The cluster factor of the disk must be 1.
- The free space on the disk cannot be fragmented into more than 100 contiguous extents.
- The index file cannot be extended.
- The master file directory (MFD) cannot be extended.

Volumes that are to be used for sequential disk save sets should be either freshly initialized or should contain only save sets. Volumes that have been used for general file processing are usually unsuitable for use with sequential disks. Sequential disk format is designed for handling relatively large save sets on relatively small disks. In a situation in which a large number of small save sets are to be placed on a disk, you should use Files-11 disk save sets (see Section 6.4.3.1).

You can read a sequential disk save set in either of two ways: as a sequential disk save set (mounted with the /FOREIGN qualifier) or as a normal Files-11 save set.

If a save set is read as a sequential disk save set, one volume can be mounted at a time (as in creating a sequential disk save set). The default directory for the save-set file specification is the directory [000000] (the MFD on the disk).

If a save set is read as a normal Files-11 save set, all volumes of the save set must be mounted (as is true for any Files-11 volume set). The default directory is your process default directory, so to read the save set you must specify the directory [000000].

Save-set files that used the file system for write operations to a disk, can also be read in sequential mode. You must specify the directory from which the save set is to be read. This mode of operation is useful when you are using standalone BACKUP. If the save set was written to a volume set using Files-11 disk structure, you must observe the following rules:

- All volumes of the volume set must be mounted. You must mount the volumes with the /FOREIGN qualifier. (If you are using standalone BACKUP, the volumes are implicitly mounted.)
- When you specify the volumes in the input specifier, you must specify the names of the devices in the same order as the relative volume number of the volumes mounted on those devices. For example, the volumes named USER01 and USER02 in a volume set are mounted on two drives, DRA3 and DRA5. The save-set specifier for the volume set must be DRA3:[directory]save-set-name.ext;v, DRA5:.

---

### 6.3.4 Remote Files-11 Save Sets

You can create or access a save set on a remote node by specifying the node name in the save-set specifier. The save set must be located on a publicly accessible disk on the remote node. Depending on the volume and file protection at the remote node, you may need to specify an access control string, which includes the login information of USERNAME and PASSWORD (for more information on access control strings, see the *VAX/VMS DCL Concepts Manual*).

---

### 6.3.5 Rotating Backup Sets

If you use disks for backups, you can make use of a "rotating backup set," in which several disks or sets of disks are used in rotation on the system. At the end of each period of use (for example, once a month), the volume or volume set currently in use is copied to the oldest set of disks, the current volume is retired, and the new copy is put on line for use during the next period.

You will find that there are advantages and disadvantages to rotating backup sets.

A rotating backup set offers two major advantages:

- 1 Your backup copy (the volume or volume set just retired) is known to be good, since it has been in use. The integrity of the new copy will be confirmed by its subsequent use; any defects discovered can be repaired using the backup copy.
- 2 The free space on the new volume is compressed, and all the files on it are made contiguous or almost contiguous, resulting in better file system performance.

There are four disadvantages to a rotating backup set:

- 1 Rotating backup sets are more vulnerable to disk errors than sets created by retiring the copy and continuing to use the original. A disk error during the copy operation results in corrupted data on your new volume; disk errors in directories or file headers will result in the loss of one or more files. Thus, you must monitor the copy operation very carefully for errors and manually repair any problems that arise.
- 2 You cannot perform the copy operation while users are updating files. The volume or volume set must be write-locked so that the copy will be consistent. This restriction only applies to rotating backup sets; it does not apply when you make a backup copy that will be retired for backup use. (Even though individual files may be incomplete, they will be covered by the next incremental backup.) For examples of how to perform and restore from incremental backups, refer to the *VAX/VMS System Manager's Reference Manual*.
- 3 Files created with explicit placement lose their placement when the volume is copied. This means you should not use a rotating backup set if the volume's primary contents are a set of database files that were carefully placed for optimized performance.
- 4 Rotating backup sets are expensive to apply to fixed-media disks (as opposed to removable media).

---

### 6.4 Performing BACKUP Tasks

This section describes how to perform the various backup tasks that are commonly used in daily system operations: copying, saving, restoring, listing, and comparing files or volumes. Examples are included for each kind of task. For an overview of these types of operations, see Section 6.2.2.

## 6.4.1 Copying Disk Files, Directories, and Volumes

When you use BACKUP to copy disk files, directories, or volumes, BACKUP creates a functional equivalent of the original. The backup copies are identical to the originals, including revision dates and protection.

Note that the BACKUP copy is not supported for DECnet operations.

### 6.4.1.1 Copying Single Files with BACKUP

You can create a backup copy of a single file by entering a command in the format:

BACKUP filespec filespec

For example, to make a backup copy of your EDTINI.EDT file and place it in a directory named [HARRIS.MISC], you would enter the following command:

```
$ BACKUP EDTINI.EDT [HARRIS.MISC]
```

This command places a duplicate of EDTINI.EDT in the directory [HARRIS.MISC]. When you specify only the directory name, BACKUP assumes a wildcard value for the file name and retains the original.

You can rename the file during the copy by specifying the new file name as you would with the DCL command COPY. For example:

```
$ BACKUP/LOG EDTINI.EDT [HARRIS.MISC]OLDEDTINI.EDT
%BACKUP-S-CREATED, created DISK$DEFAULT:[HARRIS.MISC]OLDEDTINI.EDT;4
```

This command effectively renames EDTINI.EDT by copying it to a file named OLDEDTINI.EDT in the directory [HARRIS.MISC]. The /LOG qualifier directs BACKUP to display information about the processed file. Note that the copied file retains the version number of the original, even though it is the only file processed. If you do not include a version number with the input file, BACKUP processes all existing versions of the file and applies their version numbers to the respective output files. In this case, only one EDTINI.EDT file exists, with a version number of 4.

### 6.4.1.2 Copying Multiple Files with BACKUP

You can make backup copies of multiple files in a single command line. The output specification must be a directory or wildcard character; it cannot be a list or a single file name. For example:

```
$ BACKUP/LOG ELEMENT.DAT,MATERIAL.DAT,SUBSTANCE.DAT [.STUFF]
%BACKUP-S-CREATED, created DISK$DEFAULT:[HARRIS.STUFF]ELEMENT.DAT;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[HARRIS.STUFF]MATERIAL.DAT;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[HARRIS.STUFF]SUBSTANCE.DAT;1
```

All files in the input specifier are copied to the named output directory. If the output specification had attempted to rename the files by giving three new file names, the operation would have failed and BACKUP would have issued an error message. The operation would also have failed if the output specifier had attempted to concatenate the input list into one file.

### 6.4.1.3 Copying an Entire Directory Tree with BACKUP

You can copy an entire directory tree by using the following format:

BACKUP [directory...] [directory...]

If the specified output directory does not exist, BACKUP creates it. All files and subdirectories represented by the ellipses (...) are copied from the input directory specification to the output directory specification. For example:

```
$ BACKUP/LOG [HOT.STUFF...] [OLD.STUFF...]
%BACKUP-S-CREDIR, created directory DISK$DEFAULT:[OLD.STUFF]
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF]ELEMENT.LIS;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF]ESSENCE.LIS;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF]MATERIAL.LIS;1
%BACKUP-S-CREDIR, created directory DISK$DEFAULT:[OLD.STUFF.MORESTUFF]
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.MORESTUFF]MATTER.DAT;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.MORESTUFF]NOMATTER.DAT;1
%BACKUP-S-CREDIR, created directory DISK$DEFAULT:[OLD.STUFF.MORESTUFF.WHAT]
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.MORESTUFF.WHAT]GOO.YUK;2
%BACKUP-S-CREDIR, created directory DISK$DEFAULT:[OLD.STUFF.NOMORESTUFF]
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.NOMORESTUFF]NUTHIN.LIS;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.NOMORESTUFF]ZILCH.DAT;1
%BACKUP-S-CREDIR, created directory DISK$DEFAULT:[OLD.STUFF.NOMORESTUFF.REALLY]
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF.NOMORESTUFF.REALLY]ZIP.ZAP;1
%BACKUP-S-CREATED, created DISK$DEFAULT:[OLD.STUFF]SUBSTANCE.LIS;1
```

BACKUP creates the directory [OLD.STUFF] and copies all files and subdirectories in [HOT.STUFF] to the new directory. It also creates the subdirectories and copies the corresponding files into them.

### 6.4.1.4 Copying a Disk Volume to a Disk with BACKUP

If you want to copy an entire disk volume to another disk volume, you should use the /IMAGE qualifier and specify only device names, as follows:

```
$ MOUNT/FOREIGN DBA2:
$ BACKUP/IMAGE DBA1: DBA2:
```

The /FOREIGN qualifier is necessary when you mount the output volume for an image copy operation; BACKUP initializes the volume using the initialization data from the input volume. All the files on DBA1: are copied to DBA2:, creating a functionally equivalent copy of the entire volume.

To use the /IMAGE qualifier, you need write access to both the volume index file (INDEXF.SYS) and the bit map file (BITMAP.SYS); as an alternative, the input medium may be write-locked.

---

### 6.4.1.5 Copying a Disk Volume Set to a Disk with BACKUP

To copy a disk volume set, you use the name of the volume set as the input specifier. The output specifier must include a device specification for each volume in the set. The following example creates an image copy of a two-disk volume set:

```
$ MOUNT/FOREIGN DBA1:,DBA2:
$ BACKUP/IMAGE WORK$ DBA1:,DBA2:
```

---

### 6.4.2 Saving Disk Files, Directories, and Volumes to Magnetic Tape

You must mount magnetic tapes with the /FOREIGN qualifier in order to use them in any BACKUP operation. (For detailed information, see the *VAX/VMS Mount Utility Reference Manual*.) You do not need to initialize magnetic tape volumes; note, however, that if you plan to use BACKUP with a fresh magnetic tape, you will need to initialize the magnetic tape first by including the /REWIND qualifier with the BACKUP command. As an alternative, you can use the DCL command INITIALIZE before mounting the magnetic tape.

If you have more than one magnetic tape drive available, you can use multiple magnetic tape volumes to save or restore data. If you have a large number of disk volumes to process, you may find that the multivolume magnetic tape feature is very useful for minimizing the time it takes to perform a save or restore operation. Once a magnetic tape volume is full, BACKUP rewinds it while writing to the continuation volume on the second drive. You can then mount a fresh magnetic tape on the first drive; when the second volume is full, BACKUP immediately resumes processing on the first drive.

If you want to mount multiple tape volumes, you can accomplish the task with a single command by separating the device names with a comma. For example:

```
$ MOUNT/FOREIGN MTAO:,MTA1:,MTA2:
```

---

#### 6.4.2.1 Saving Disk Files to Magnetic Tape

You can create a save set on magnetic tape of a single disk file by entering a command in the following format at the DCL prompt:

```
$ BACKUP filename save-set-specifier
```

The save-set specifier can include any file type that you like; typically, the BCK extension is used to identify the file as a save set. Note, however, that save-set names on magnetic tape are limited to 17 characters (see Section 6.3.1). For example, you might save the disk file CONTRACTS.DAT as follows:

```
$ BACKUP CONTRACTS.DAT MTAO:CONTRACTS.BCK
```

To save a list of files, separate the file names with commas as in the following example:

```
$ BACKUP PAPERFILE,DBAO:[METALFILE]NAILFILE.DAT MTA1:MYFILES.BCK
```

Note that by default BACKUP saves all versions of each file. If you want to save only the most recent versions of files in a directory, use the following format:

```
BACKUP [directory]*.*; save-set specifier
```

For example:

```
$ BACKUP [AUDIOFILE...]*.*; MTAO:MYPHILE.BCK
```

BACKUP saves the most recent version of each file in the directory [AUDIOFILE], as well as the most recent version of each file in its subdirectories.

---

#### 6.4.2.2 Saving Directories to Magnetic Tape

You can create a save set of the current default directory and all its contents as follows:

```
$ BACKUP * MTAO:OCT12SAVE.BCK
```

All the files in the current default directory will be saved in a save set named OCT12SAVE.BCK (a reminder that the save operation was done on October 12). This example assumes that your target tape is mounted on device MTA0:.

---

#### 6.4.2.3 Saving a Directory Tree to Magnetic Tape

To create a save set that contains all files and subdirectories of a directory named [ACCOUNTS], enter the following:

```
$ BACKUP [ACCOUNTS...] MTA0:OCT12ACC.BCK
```

All files in the directory [ACCOUNTS], all subdirectories of [ACCOUNTS], and all the files in those subdirectories will be included in the save set OCT12ACC.BCK. For more information on the use of BACKUP wildcard characters, see the *VAX/VMS Backup Utility Reference Manual*.

---

#### 6.4.2.4 Saving Disk Volumes to Magnetic Tape

If you wish to save an entire disk volume, you should use the /IMAGE qualifier in the following format:

BACKUP/IMAGE ddcu: save-set-specifier

The image save operation saves all files on the input volume; therefore, the /IMAGE qualifier is incompatible with other file selection qualifiers. Note that if a file is entered in more than one directory on the input volume, the file is saved only once.

You can use the /IMAGE qualifier to save a disk volume to a multivolume save set on more than one tape drive. For example:

```
$ BACKUP/IMAGE DBA0: MTA1:20JUL.BCK,MTA2:/REWIND
```

You specify the save-set name only in the first element of the output-specifier list. Assuming that you have mounted the two magnetic tape volumes (as discussed in Section 6.4.2), BACKUP automatically continues the save operation on MTA2: after MTA1: is full.

---

#### 6.4.2.5 Saving Unstructured Disk Volumes to Magnetic Tape

If the disk volume you want to save is unstructured, use the /PHYSICAL qualifier.

```
$ BACKUP/PHYSICAL DMA0: MTA2:PHYSBACK.BCK
```

The input volume DMA0: is saved in terms of logical blocks. Note that the resulting save set PHYSBACK.BCK can only be restored with a physical restore operation (using the /PHYSICAL qualifier).

### 6.4.3 Saving Disk Files, Directories, and Volumes to Disks

You can write save sets to a disk in either of two modes: standard Files-11 disk structure, or sequential. In either case, you must use the output save-set qualifier `/SAVE_SET`. The `/SAVE_SET` qualifier indicates to BACKUP that you want the output disk to contain a save set rather than a copy of the selected files.

The format for saving a disk file to a disk is as follows:

```
$ BACKUP filespec ddcu:save-set-name/SAVE_SET
```

However, the procedure that is appropriate for mounting the output disk depends upon whether the save set is to be written in standard Files-11 format (file structured) or sequential format.

#### 6.4.3.1 Writing Save Sets on File-Structured Disk Volumes

To write save sets on a file-structured disk, mount the disk as follows:

```
$ MOUNT ddcu volume-label
```

For more information on the VAX/VMS Mount Utility, refer see *VAX/VMS Mount Utility Reference Manual*.

You can use the following commands to mount a standard Files-11 disk on drive DBA2: and save the file `CONTRACTS.DAT` to a save set named `CONTRACTS.BCK` in the directory `[SAVE]`:

```
$ MOUNT DBA2 volume-label
$ BACKUP CONTRACTS.DAT DBA2:[SAVE]CONTRACTS.BCK/SAVE_SET
```

The qualifier `/SAVE_SET` is necessary for writing the output file in BACKUP format. Without the `/SAVE_SET` qualifier, the output file would be a copy of the original.

#### 6.4.3.2 Writing Save Sets on Sequential Disk Volumes

As an alternative to standard Files-11 structure, you can write save sets to a disk sequentially, as they would be written on magnetic tape. The primary advantage in using sequential disk save sets is that you can treat the disk volume as you would a magnetic tape save set; this flexibility allows you to mount multivolume save sets one volume at a time. See Section 6.4.3.3 for more information on multivolume save sets.

To save data to a sequential disk save set, make certain that the output specifier includes a device name for a disk device, a save-set name, and the qualifier `/SAVE_SET`; do not include a directory name.

To write sequential disk save sets, you must mount the output disk with the /FOREIGN qualifier. Although the volume is mounted foreign, BACKUP manages the disk using Files-11 structure.

Volumes that are to be used for sequential disk save sets should be either freshly initialized or should contain only save sets. Volumes that have been used for general file processing are usually unsuitable for sequential disk use. You can initialize the volume by using the BACKUP qualifier /INITIALIZE. For example, you can use the following commands to mount and initialize a sequential disk on drive DMA0: and save the file CONTRACTS.DAT to a save set named CONTRACTS.BCK:

```
$ MOUNT/FOREIGN DMA0:
$ BACKUP/LOG CONTRACTS.DAT DMA0:CONTRACTS.BCK/SAVE_SET/INITIALIZE-
_$ /LABEL=SAVEWORK
%BACKUP-S-COPIED, copied DBA2:[USER]CONTRACTS.DAT;1
```

The MOUNT/FOREIGN command tells BACKUP to write to the output volume sequentially. The /SAVE\_SET qualifier on the BACKUP command output specifier is necessary for writing the output file in BACKUP format. The /INITIALIZE qualifier destroys any previous structure information on the disk, enables the volume to be overwritten, and assigns the volume label SAVEWORK.

When you initialize the volume, the volume label you specify is permanent; that is, the label remains in effect until either of the following takes place:

- You change it with the DCL command SET VOLUME/LABEL.
- The volume is reinitialized.

See the *VAX/VMS Backup Utility Reference Manual* for more information on BACKUP/INITIALIZE. See the *VAX/VMS DCL Dictionary* for more information on SET VOLUME.

### 6.4.3.3 Writing Multivolume Sequential Disk Save Sets

You may find the multivolume option particularly useful if your system has no tape drive but does have a large fixed-media disk and a small removable disk. When one sequential disk is full, BACKUP prompts you for another disk, as it would for save sets on magnetic tape. You can use more than one disk device at a time to save or restore data; this allows processing to continue on another disk while the one most recently used is spinning down. Note that you need the user privilege LOG\_IO to read or write a multivolume sequential disk save set.

BACKUP normally does not initialize the first sequential disk volume, as the default is /NOINITIALIZE; however, continuation volumes are always initialized. You can initialize the first volume of a sequential disk save set by specifying the /INITIALIZE qualifier.

### 6.4.4 Backing Up Full Volumes and Volume Sets

If you wish to back up an entire disk volume or volume set, you should use the /IMAGE qualifier. You can perform an image backup in a copy operation (disk to disk, as described in Section 6.4.1) or in a save operation. When you use the /IMAGE qualifier in a save operation, BACKUP creates a save set that contains data necessary for reinitializing the disk volume. (The attribute record that contains this information is called the save-volume summary record.)

When you use the /IMAGE qualifier in a copy or restore operation, the output volume is reinitialized and the restored volume is a functionally equivalent copy of the original volume. If the output volume has already been initialized as a Files-11 volume, you can use the /NOINITIALIZE qualifier to reinitialize the volume using the volume initialization parameters found on the volume.

You cannot use other file selection qualifiers with the /IMAGE command qualifier. All files on the disk are saved; this includes reserved files and lost files (files that have no directory entry).

The following example shows a save operation from a Files-11 disk to magnetic tape:

```
$ BACKUP/IMAGE DRA1: MTAO:1JUN.BCK
```

If you use magnetic tape as the backup medium, you may need to mount additional magnetic tapes. The number of magnetic tapes depends on the size of the disk being saved and the amount of space in use on the disk.

If you use disk as the backup medium, you can use BACKUP to either copy files to the new disk or to create a save set on the new disk. If you create a save set on the new disk, you must create a directory to which the save set will be written and you must use the /SAVE\_SET output save-set qualifier. The directory must be included in the save-set name or it must be your default directory.

For example:

```
$ SHOW DEFAULT  
DMA1: [SYSTEM]  
$ BACKUP/IMAGE DMA1: DB4: [BACKUPS]21JANDMA1.BCK/SAVE_SET
```

You should specify the /RECORD command qualifier if you are performing the full volume backup in conjunction with incremental backups that use the /RECORD qualifier. (For more information on incremental backups, refer to the *VAX/VMS System Manager's Reference Manual*.)

You can back up an entire volume set by following the same procedures outlined for backing up a disk volume. Simply name the device on which the root volume (volume number 1) is mounted.

### 6.4.4.1 Backing Up a Disk Volume Set When Drives are Limited

When you use BACKUP with a volume set, you must mount all volumes of the set. Therefore, it may not be possible to copy a large volume set directly to a disk if you have a limited number of disk drives. The following example shows how you can copy the volume set one volume at a time with the BACKUP/IMAGE/VOLUME command, using one more drive than the number of volumes in the volume set. You must write-lock the volume set during the entire procedure to ensure consistency.

(suitable warning broadcasts)

```
$ DISMOUNT/NOUNLOAD DRA0:
$ MOUNT/SYSTEM/NOWRITE DRA0:,DRA1:,DRA2: PUBLIC01,PUBLIC02,PUBLIC03
%MOUNT-I-MOUNTED, PUBLIC01 mounted on _DRA0:
%MOUNT-I-MOUNTED, PUBLIC02 mounted on _DRA1:
%MOUNT-I-MOUNTED, PUBLIC03 mounted on _DRA2:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH01 mounted on _DRA3:
$ BACKUP/IMAGE/VOLUME=1 DRA0: DRA3:
$ DISMOUNT DRA3:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH02 mounted on _DRA3:
$ BACKUP/IMAGE/VOLUME=2 DRA0: DRA3:
$ DISMOUNT DRA3:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH03 mounted on _DRA3:
$ BACKUP/IMAGE/VOLUME=3 DRA0: DRA3:
$ DISMOUNT DRA3:
$ DISMOUNT/NOUNLOAD DRA0:
$ MOUNT/SYSTEM DRA0:,DRA1:,DRA2: PUBLIC01,PUBLIC02,PUBLIC03
%MOUNT-I-MOUNTED, PUBLIC01 mounted on _DRA0:
%MOUNT-I-MOUNTED, PUBLIC02 mounted on _DRA1:
%MOUNT-I-MOUNTED, PUBLIC03 mounted on _DRA2:
```

(announce public disk is available again)

The operator needs to back up a three-volume set. Thus, at least four drives are required. The operator warns the users that drives DRA0:, DRA1:, and DRA2: are about to be WRITE-LOCKED for backups. Next the operator issues MOUNT commands for the volumes PUBLIC01, PUBLIC02, and PUBLIC03 on drives DRA0:, DRA1:, and DRA2:, respectively, to specify write-locking with the /NOWRITE qualifier. Note that PUBLIC01 is the root volume. A scratch volume is mounted on drive DRA3: to be used for the target volumes. As the volume on DRA3: is filled, it is dismounted and a new scratch volume is mounted. This procedure repeats until the third volume has been copied. Then the last volume on DRA3: is dismounted. To enable writing on the volume set again, the operator dismounts (but does not unload) DRA0:, then mounts the volumes again, omitting the /NOWRITE qualifier. As a final step, the operator announces to the users that the volume set is available again for writing.

### Note

**Do not simply WRITE-LOCK a mounted volume; doing so is likely to cause errors and/or suspension of system activity. You must dismount the public volume set, remount it write-locked, and continue as described previously. This is critical to the success of the operation.**

## 6.4.5 Performing Selective Backups

You may find selective backups necessary for certain groups of files that require special treatment. Generally, if files must be backed up regularly, you should create a command procedure that contains the required backup commands.

For more information on creating command procedures, see Section 6.6 of this guide or see the *Guide to Using DCL and Command Procedures on VAX/VMS*.

One way to perform selective backups is through the use of wildcard characters. For example, you can use the asterisk wildcard character (\*) to select files by criteria such as file name, file type, or version number. This command selects files by version number and copies them to the directory [SAVE] on a specified RK07 drive:

```
$ BACKUP/LOG *.*;3 DMA1:[SAVE]
%BACKUP-S-CREATED, created DMA1:[SAVE]CHARTS.DAT;3
%BACKUP-S-CREATED, created DMA1:[SAVE]FIGURES.DAT;3
%BACKUP-S-CREATED, created DMA1:[SAVE]LISTS.DAT;3
%BACKUP-S-CREATED, created DMA1:[SAVE]TABLES.DAT;3
```

The commands in the following example copy files from a directory and a directory tree on DMA0: to newly created directories on DMA1:. All files in the [GEORGE] directory and its subdirectories on the source disk are copied to directories with the same name on the target volume; the /OWNER\_UIC=ORIGINAL qualifier ensures that the output files retain the owner UICs of the input files. All files with the file name DUNGEON in the [SYSEXE] directory on the source disk are copied to the [SYSEXE] directory on the target volume.

```
$ BACKUP DMA0:[GEORGE...] DMA1:[*...]/OWNER_UIC=ORIGINAL
$ BACKUP DMA0:[SYSEXE]DUNGEON.*;* DMA1:[SYSEXE]
```

BACKUP has qualifiers that allow you to select files according to criteria such as UIC, creation date, expiration date, and modification date. For example, you can select files according to UIC by using the /OWNER\_UIC qualifier as follows:

```
$ BACKUP/LOG [.MEM]/OWNER_UIC=[300,16] $1$DLA2:[USER]
%BACKUP-S-CREATED, created $1$DLA2:[USER]ASSIGN.LIS;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]BBALL.DIS;19
%BACKUP-S-CREATED, created $1$DLA2:[USER]BBLEAGUE.FISTATEMENT;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]BRULES.LIS;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]GYM.MAI;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]INFO.DAT;2
%BACKUP-S-CREATED, created $1$DLA2:[USER]MEMO.MAI;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]NEWS.MAI;1
%BACKUP-S-CREATED, created $1$DLA2:[USER]TEAM.LIS;19
```

In this example, all files residing in the subdirectory [.MEM] with an owner UIC of [300,16] are processed. Note that the brackets are required when you specify the UIC code. You can specify the UIC of the current default process by simply entering the /OWNER\_UIC qualifier without a code.

If you want to select your input files according to their date of creation, you can use the /CREATED qualifier with /BEFORE or /SINCE. The following command copies all files in the directory [TRUBBLE] that were created on or since April 3, 1985:

```
$ BACKUP DMA1:[TRUBBLE]*/SINCE=03-APR-1985/CREATED DBA2:[SPRS]
```

You can use the /EXPIRED qualifier with /SINCE or /BEFORE to select files according to the expiration date recorded in the file header.

```
$ BACKUP *.COB;*/BEFORE=20-JUL-1985:/EXPIRED MFA0:JUL20SAVE.BCK
```

In this example, all versions of all files in the current default directory, with a file type of .COB and an expiration date earlier than July 20, 1985, will be saved to JUL20SAVE.BCK on MFA0:.

In some cases, you may find it more practical to exclude certain files from a backup operation. You can do this as in the following example:

```
$ BACKUP DBA2:[OSCAR...]/EXCLUDE=(*.LOG;*.NOTES.*;*) MTA1:SAVE.BCK
```

This command directs BACKUP to save all the files in the directory [OSCAR] and its subdirectories, except for those with a file type of LOG and those with the file name NOTES, regardless of file type or version number. The parentheses are required when a list is specified with the /EXCLUDE qualifier.

### 6.4.6 Protecting a BACKUP Save Set

You can use the output save-set qualifiers /OWNER\_UIC and /PROTECTION to protect save sets on magnetic tape, sequential disk, or Files-11 disk. Specifying the protection and owner UIC for a save set can prevent nonprivileged users from mounting a save-set volume or accessing a Files-11 save set (anyone who has access to a save set can read any data contained in the save set).

For example:

```
$ BACKUP [HOOVER...] MFA2:PINKERTON.BCK/OWNER_UIC=[300,99]/REWIND
```

The directory tree [HOOVER...] is saved to PINKERTON.BCK on the magnetic tape drive MFA2: The output qualifier /OWNER\_UIC assigns an owner UIC of [300,99] to the save-set volume.

Because the protection on magnetic tape is encoded in the volume label, the /OWNER\_UIC and /PROTECTION qualifiers are effective on magnetic tape save sets only if you specify the /REWIND qualifier (to initialize the magnetic tape).

```
$ BACKUP [PAYROLL] MFA2:KNOX.BCK/OWNER_UIC=[3,3] -  
_$/PROTECTION=(S:RWE,O:RWED,G:RE,W)
```

In this example, the directory [PAYROLL] is saved to KNOX.BCK on the magnetic tape drive MFA2:. The /OWNER\_UIC qualifier assigns an owner UIC of [3,3] to the save set. The /PROTECTION qualifier assigns the owner of the save-set volume READ, WRITE, EXECUTE, and DELETE access. SYSTEM users are assigned READ, WRITE, and EXECUTE access; GROUP users are assigned READ and EXECUTE access; and WORLD users are assigned no access.

## 6.4.7 Restoring Disk Files, Directories, and Volumes from Save Sets

The format for restoring from a save set is as follows:

BACKUP save-set-specifier[/SAVE\_SET] output-specifier

You use the /SAVE\_SET qualifier when the save-set specifier refers to a disk volume. The output specifier can be a device, directory, file name, or wildcard character.

The following sections describe how to perform various restore operations.

### 6.4.7.1 Restoring Disk Files from Save Sets on Magnetic Tape

You can restore all files in the save set to the current default directory as shown in the following example:

```
$ BACKUP TAPE:NOV12SAVE.BCK []
```

The following command restores files to a subdirectory tree:

```
$ BACKUP TAPE:NOV12SAVE.BCK [LYKINS...]
```

If you want to restore a specific file from a save set, you can use the /SELECT qualifier. For example:

```
$ MOUNT/FOREIGN MTAO:
%MOUNT-I-MOUNTED, NOV25A mounted on _MTAO:
$ BACKUP
_From: MTAO:NOV2SAVE.BCK/SELECT=[LYKINS.GLENDO]STRAT1.DAT;5
_To:   STRAT1.DAT;5
$ DIRECTORY STRAT1.DAT
Directory [LYKINS.GLENDO]
STRAT1.DAT;5
Total of 1 file.
$
```

The file STRAT1.DAT in the directory [LYKINS.GLENDO] has been accidentally deleted. The user had previously saved the file in the save set NOV2SAVE.BCK and now restores it to its original directory by using the /SELECT qualifier.

If you omit a save-set name for an input magnetic tape, BACKUP reads the first save set it encounters on the magnetic tape and stops processing when it reaches the end of that save set. Since BACKUP does not automatically rewind until it reaches the EOT, you can proceed to the next save set (if any exists) by repeating the command. As an alternative, you can include an asterisk (\*) with the device specification, which directs BACKUP to read all of the save sets on the magnetic tape volume.

#### 6.4.7.2 Restoring Disk Files from Save Sets on Sequential Disk

You can read a sequential disk save set in either of two ways: as a sequential disk save set (mounted with the /FOREIGN qualifier) or as a normal Files-11 save set.

If you choose to read the save set as sequential, you can mount one volume at a time (as you can do when you create a sequential disk save set). The default directory for the save set file specification is the directory [000000], which is the master file directory (MFD) on the disk; therefore, you do not need to include the directory in the input specification. For example:

```
$ MOUNT/FOREIGN DMAO:
$ BACKUP/LOG DMAO:CONTRACTS.BCK/SAVE_SET CONTRACTS.DAT
%BACKUP-S-CREATED, created DBA2:[USER]CONTRACTS.DAT;1
```

Since the volume is mounted foreign, the DCL command DIRECTORY would fail and generate an error message; however, BACKUP recognizes the format and expects to find the save set CONTRACTS.BCK in the MFD directory. The /SAVE\_SET qualifier is required on any operation that restores from a disk.

As an alternative, you can read the save set as a normal Files-11 volume. If the save set is multivolume, you must mount all volumes of the set (as you must for any Files-11 volume set).

```
$ MOUNT DMAO: SAVEDWORK
$ BACKUP/LOG DMAO:[000000]CONTRACTS.BCK/SAVE_SET CONTRACTS.DAT
%BACKUP-S-CREATED, created DBA2:[USER]CONTRACTS.DAT;1
```

The default directory is your process default directory, so you must specify the directory [000000]. The SAVE\_SET qualifier is necessary for restoring the file to its original structure; without the /SAVE\_SET qualifier, the save set would simply be copied.

Save-set files written to a disk using the file system can also be read in sequential mode; you may find this mode of operation useful if you use standalone BACKUP. You must specify the directory from which the save set is to be read. If the save set was written to a volume set using Files-11 disk structure, you must observe the following rules:

- All volumes of the volume set must be mounted. You must mount the volumes with the /FOREIGN qualifier. (If you are using standalone BACKUP, the volumes are implicitly mounted.)

- When you specify the volumes in the input specifier, you must specify the names of the devices in the same order as the relative volume numbers of the volumes mounted on those devices. For example, the volumes named USER01 and USER02 in a volume set are mounted on two drives, DRA3: and DRA5:. The save-set specifier for the volume set must be as follows:  
DRA3:[directory]save-set-name.ext;v, DRA5:

### 6.4.7.3 Restoring Disk Files from Save Sets on Multiple Volumes

If your save set encompasses more than one tape or sequential disk volume, it is possible to begin restore operations without mounting the first volume of the save set. However, if you use the /IMAGE qualifier, processing must begin with the first volume. If the tape that you mount is not the first volume in the save set, you will receive the warning message:

```
%BACKUP-W-NOT1STVOL, tape 'name' is not the start of a save set
```

If you restore a small number of files from a large save set, it is a good idea to use the /LOG qualifier. This enables you to monitor the files as they are restored. The BACKUP command will continue processing until it reaches the end of the save set. Once the files you need have been restored, you can press CTRL/Y to terminate processing.

### 6.4.8 Restoring Entire Disk Volumes

To restore an entire disk volume from a save set that was created using the /IMAGE command qualifier, you must mount the new volume using the DCL command MOUNT/FOREIGN. You must then restore the volume with the BACKUP/IMAGE command.

When a save set is created using the /IMAGE command qualifier, the data necessary for reinitializing the volume is placed in the save set. When the /IMAGE command qualifier is used to restore the volume, the new volume is initialized using that data in the save set.

For example:

```
$ MOUNT/FOREIGN DRA1
%MOUNT-I-MOUNTED, 24JUND mounted on _DRA1:
$ BACKUP/IMAGE MTA0:24JUNDMA1.BCK DRA1:
```

The volume is initialized using the initialization parameters saved in the save set. All files and directories in the save set are restored to the new volume.

To restore a volume set in image mode, you must mount all the volumes of the set as foreign volumes. You must also specify the list of devices on which the volumes are mounted in the output-specifier of the BACKUP command.

```
$ MOUNT/FOREIGN DRA0:  
%MOUNT-I-MOUNTED, PUBLIC01 mounted on _DRA0:  
$ MOUNT/FOREIGN DRA1:  
%MOUNT-I-MOUNTED, PUBLIC02 mounted on _DRA1:  
$ MOUNT/FOREIGN DRA2:  
%MOUNT-I-MOUNTED, PUBLIC03 mounted on _DRA2:  
$ BACKUP/IMAGE MTAO:31JUNPUB DRA0:,DRA1:,DRA2:
```

The volumes receive volume set numbers in the order in which they are listed in the BACKUP command. In this example, DRA0:, DRA1:, and DRA2: become volumes 1, 2, and 3, respectively.

### 6.4.8.1 Changing Volume Initialization Parameters Before Restoring

If you wish to change the volume initialization parameters for a volume, you need to perform the following steps:

- 1 Initialize the new volume using the new initialization parameters.
- 2 Mount the new volume using the /FOREIGN qualifier.
- 3 Restore the volume with BACKUP, using the /NOINITIALIZE command qualifier.

In the following example the cluster size of the volume is being changed to 3. The other volume initialization parameters take the default values from the INITIALIZE command.

```
$ ALLOCATE DRA2  
%DCL-I-ALLOC, _DRA2: allocated  
$ INITIALIZE/CLUSTER_SIZE=3 DRA2: TEST_PROGS  
$ MOUNT/FOREIGN DRA2  
%MOUNT-I-MOUNTED, TEST_PROGS mounted on _DRA2:  
$ BACKUP/IMAGE/NOINITIALIZE/TRUNCATE MTAO:1JUN.BCK DRA2:
```

The only initialization parameter that you cannot change is the structure level of the volume. If you change the cluster factor, as in the above example, it is good practice to include the /TRUNCATE qualifier.

### 6.4.9 Listing the Contents of a BACKUP Save Set

You can use the /LIST command qualifier to list information about files in a save set, either at the terminal or to a specified output file. BACKUP can perform this operation in conjunction with any other BACKUP operation, or by itself.

The following command will list save-set information at the terminal:

```
$ BACKUP/LIST 2MAR1555.BCK/SAVE_SET
```

To list the same information in a file named INFO.LIS, enter the following command:

```
$ BACKUP/LIST=INFO.LIS 2MAR1555.BCK/SAVE_SET
```

If you do not know the save set names on a magnetic tape volume, you can specify the device alone. Note, however, that BACKUP then performs the list operation as it would a restore operation with only a device specification: It reads the first save set it encounters on the magnetic tape and stops processing when it reaches the end of that save set. BACKUP does not automatically rewind until it reaches the EOT (unless you have specified the /REWIND qualifier); therefore, you can proceed to the next save set (if any exists) by repeating the command.

By including an asterisk (\*) with the device specification, you can direct BACKUP to list all of the save sets on the volume:

```
$ BACKUP/LIST MTAO:*
```

If you want to list only the names of the save sets on a volume, without the names of the files in the save sets, you need to mount the magnetic tape volume as Files-11; then use the DCL command DIRECTORY. For example:

```
$ MOUNT MTAO: SAVEDWORK TAPE
%MOUNT-I-MOUNTED, SAVEDWORK mounted on _MTAO:
$ DIRECTORY/SIZE/DATE/PROTECTION TAPE:
Directory MTAO: []
CONTRACTS.BCK;1          5  13-JUN-1985 12:11 (RWED,RWED,RE,.)
JUL20SAVE.BCK;1          3  20-JUL-1985 23:59 (RWED,RWED,RE,.)
MYPHILE.BCK;1            2  28-SEP-1985 12:00 (RWED,RWED,RE,.)
Total of 3 files, 10 blocks.
```

### 6.4.10 Comparing Files with BACKUP

You can use the BACKUP compare operation either to compare a save set with files on a disk, or to compare files on a disk with other files on a disk. Note that the /COMPARE qualifier should be used only to compare a save set against original files or to compare files or volumes copied using BACKUP. BACKUP processes files by blocks. Comparing files that were not produced by BACKUP may cause errors.

You can compare a save set with files on a disk as in the following example (this command directs BACKUP to compare the contents of the save set 2MAR1555.BCK with the directory [LYKINS]):

```
$ BACKUP/COMPARE TAPE:2MAR1555.BCK [LYKINS]
```

The following example compares files on a disk; note that in this example, there is an inconsistency in block 16 between UPLIFT.EXE;4 and UPLIFT.EXE;3:

```
$ BACKUP/COMPARE UPLIFT.EXE;3 UPLIFT.EXE;4
%BACKUP-E-VERIFYERR, verification error for block 16 of WRKD$: [LYKINS]UPLIFT.EXE;4
```

---

## 6.5 Using BACKUP Journal Files

A BACKUP journal file contains records of BACKUP save operations and the file specifications of the files that were saved with each operation. To find a particular file in a multivolume save set, you can review the BACKUP journal file to find the magnetic tape volume that contains the file.

Use the /JOURNAL qualifier to create, or append information to, a BACKUP journal file. If no file specifier appears with the /JOURNAL command qualifier, the name of the BACKUP journal file defaults to SYS\$DISK:BACKUP.BJL. The default file type for BACKUP journal files is BJL.

If the specified BACKUP journal file does not exist, it is created; if it already exists, the new journal information is appended to the existing journal file. You can start a new version of a BACKUP journal file by creating an empty file with an editor such as EDT or the DCL command CREATE.

To list a BACKUP journal file, enter the command:

```
BACKUP/LIST[=file-spec]/JOURNAL[=file-spec]
```

You must not specify an input or output specifier with a BACKUP/JOURNAL/LIST command. If the file specification is omitted from the /LIST qualifier, output is directed to SYS\$OUTPUT; if the file specification is omitted from the /JOURNAL qualifier, the default BACKUP journal file is used.

You can use file selection qualifiers with the BACKUP/JOURNAL/LIST command. This allows you to locate a set of files in a save set just as the DIRECTORY command allows you to locate a set of files on a disk. The following example shows all files in the directory [SMITH.PROGS] backed up after October 5, 1985, listed in the BACKUP journal file ARCH.BJL:

```
$ BACKUP/LIST/JOURNAL=ARCH.BJL/SELECT=[SMITH.PROGS]/SINCE=5-OCT-1985
```

## Maintaining Data Integrity on Disk and Magnetic Tape Volumes

The following example shows the use of BACKUP journal files:

```
$ BACKUP/JOURNAL/LOG/IMAGE DRA2: MTA0:3OCT.FUL
```

(first set is printed here)

```
%BACKUP-I-RESUME, resuming operation on volume 2
%BACKUP-I-READYWRITE, mount volume 2 on _MTA0: for writing
Press return when ready: RET
```

(second set is printed here)

```
$ BACKUP/JOURNAL/LIST
Listing of BACKUP journal
Journal file _DB2:[SYSMGR]BACKUP.BJL;1 on 3-OCT-1985 00:40:56.36
Save set 3OCT.FUL created on 3-OCT-1985 00:40:56.36
Volume number 1, volume label 3OCT01
```

```
[COLLINS]ALPHA.DAT;4
[COLLINS]EDTINI.EDT;5
[COLLINS]LOGIN.COM;46
[COLLINS]LOGIN.COM;45
[COLLINS]MAIL.MAI;1
[COLLINS]MAR.DIR;1
[COLLINS.MAR]GETJPI.EXE;9
[COLLINS.MAR]GETJPI.LIS;14
```

```
[LANE]LES.MAI;1
```

```
Save set 3OCT.FUL created on 3-OCT-1985 00:40:56.36
Volume number 2, volume label 3OCT02
```

```
[LANE]MAIL.MAI;1
[LANE]MEMO.RNO;5
[LANE]MEMO.RNO;4
```

```
[WALTERS.VI]KD.RNO;52
```

End of BACKUP journal

## 6.6 Using Command Procedures to Perform Backup Operations

If you have specific backup operations that you perform on a regular basis, you may find that command procedures are useful. For example, if you have files that should be saved on a daily basis, you probably use the same command strings repeatedly. You can use a command procedure to issue the necessary commands for you, as in the following sample:

```
$ ON CONTROL_Y THEN GOTO EXIT
$ ON ERROR THEN GOTO EXIT
$ GET_DRIVE: INQUIRE DRIVE "Drive"
$ IF DRIVE .EQS. "" THEN GOTO GET_DRIVE
$ ALLOCATE 'DRIVE'
$ MOUNT/FOREIGN 'DRIVE'
$ GET_SAVESET_SPEC: INQUIRE SPEC "Save set name"
$ IF SPEC .EQS. "" THEN GOTO GET_SAVESET_SPEC
$ BACKUP/RECORD/IGNORE=INTERLOCK/SINCE=BACKUP [...] 'DRIVE': 'SPEC'/SAVE_SET
$ EXIT:
$ DISMOUNT 'DRIVE'
$ DEALLOCATE 'DRIVE'
```

This command procedure allocates, initializes, and mounts a specified device. It allows you the flexibility of employing either a magnetic tape or a sequential disk output volume.

The procedure also performs an incremental backup on the current directory tree, saving only the files that have been modified since the last backup operation. (For more information on incremental backups, refer to the *VAX/VMS System Manager's Reference Manual*.)

Note that if you do not have an ON ERROR statement in your command procedure and BACKUP returns a FATAL or ERROR status code, the command procedure exits immediately after the BACKUP command is executed. If you want your command procedure to complete, you should include an ON ERROR statement such as the one in this procedure.

Note also that this procedure assumes a colon is not included with the device name. If you were to include the colon, the operation would fail because the colon is given in the BACKUP command string. (The extra colon would cause BACKUP to attempt a network function, interpreting the device specification as a node name.) You can create a more sophisticated procedure that checks for the extra colon and removes it if necessary, as in the following example:

```

$ ON CONTROL_Y THEN GOTO EXIT
$ ON ERROR THEN GOTO EXIT
$ WRITE SYS$OUTPUT "Welcome to FETCH."
$ WRITE SYS$OUTPUT " "
$ L1: INQUIRE/NOPUNC PHYS "Have you placed the volume in the drive? "
$ IF .NOT. PHYS THEN GOTO L1
$ INQUIRE/NOPUNC DRIVE "Which drive is the volume mounted on? "
$ DRIVE = DRIVE - ":"
$ ALLOCATE 'DRIVE'
$ MOUNT/FOREIGN 'DRIVE'
$ ON ERROR GOTO COMMAND_LOOP
$ !
$ COMMAND_LOOP: INQUIRE/NOPUNC OPTION "Fetch> "
$ IF OPTION .EQS. "DIR" THEN GOTO DIR
$ IF OPTION .EQS. "EXIT" THEN GOTO EXIT
$ IF OPTION .EQS. "FETCH" THEN GOTO FETCH
$ IF OPTION .EQS. "HELP" THEN GOTO HELP
$ IF OPTION .EQS. "LIST" THEN GOTO LIST
$ GOTO COMMAND_LOOP
$ !
$ DIR: INQUIRE SPEC "File spec"
$ DIR 'SPEC'
$ GOTO COMMAND_LOOP
$ !
$ HELP:
$ WRITE SYS$OUTPUT "Enter any of the following commands at the prompt:"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "DIR" (To search for a file)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "EXIT" (To exit this program)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "FETCH" (To perform a BACKUP RESTORE operation)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "HELP" (To read this text)"
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "LIST" (To perform a BACKUP LIST operation)"
$ GOTO COMMAND_LOOP
$ !
$ FETCH: INQUIRE FILE "Filespec"
$ INQUIRE SAVESET "Save set name"
$ LINE := BACKUP/LOG 'DRIVE': 'SAVESET'/SELECT='FILE'
$ INQUIRE EXCLUDE "Enter any filespecs you want excluded"
$ IF EXCLUDE .EQS. "" THEN GOTO L2
$ LINE := 'LINE'/EXCLUDE=('EXCLUDE')

```

```

$ !
$ L2: INQUIRE/NOPUNC TO "Where do you want the file(s)? (<RET> for current directory)"
$ IF TO .EQS. "" THEN GOTO REPLACE
$ LINE := 'LINE' 'TO'
$ GOTO L3
$ REPLACE: LINE := 'LINE' []
$ !
$ L3: INQUIRE/NOPUNC NEW "Create a new version if file already exists? "
$ IF .NOT. NEW THEN GOTO NOT
$ LINE := 'LINE'/NEW_VERSION
$ !
$ NOT: LINE := 'LINE'/OWNER_UIC=ORIGINAL
$ LINE
$ GOTO COMMAND_LOOP
$ !
$ LIST: INQUIRE SPEC "Filespec"
$ INQUIRE SAVESET "Save set name"
$ INQUIRE/NOPUNC OUTPUT "What do you want to call the list file? (<RET> for SYS$OUTPUT ")
$ IF OUTPUT .EQS. "" THEN GOTO NOOUT
$ LINE := BACKUP/LIST='OUTPUT' 'DRIVE': 'SAVESET'/SELECT=('SPEC')
$ GOTO L4
$ NOOUT: LINE := BACKUP/LIST 'DRIVE': 'SAVESET'/SELECT=('SPEC')
$ !
$ L4: INQUIRE EXCLUDE "Enter any filespecs you want excluded"
$ IF EXCLUDE .EQS. "" THEN GOT L5
$ LINE := 'LINE'/EXCLUDE=('EXCLUDE')
$ !
$ L5: LINE
$ GOTO COMMAND_LOOP
$ !
$ EXIT:
$ DISMOUNT 'DRIVE'
$ DEALLOCATE 'DRIVE'

```

Like the previous procedure, this sample allows you the flexibility of choosing either a magnetic tape or a disk device. However, the functions provided are quite different and more complex. After you invoke the procedure, you are in a program that displays its own prompt (Fetch> ) and has its own set of command options.

In addition to allocating and mounting the specified device, the program can perform a BACKUP restore or list operation. Whatever function you choose causes the program to prompt you with additional options.

You can tailor these procedures to suit your own needs. If you need further information on DCL command procedures, see the *Guide to Using DCL and Command Procedures on VAX/VMS*.

# 7

## Handling Operator-Oriented Disk and Magnetic Tape Functions

This chapter describes some of the disk and magnetic tape functions typically handled by VAX operators. Operator functions discussed include:

- Using the operator terminal to communicate with users
- Handling mount requests
- Handling requests from the Backup Utility
- Handling mount verification tasks

### 7.1 Using the Operator Terminal to Handle User Requests

One of the chief operator functions, communicating with users in handling requests, must be performed at terminals that have been defined as operator terminals. Operators themselves can set up an operator terminal by using the DCL command `REPLY/ENABLE` at the desired terminal. When you enter this command, the following Operator Communication Facility (OPCOM) message will be displayed at the terminal.

```
XXXXXXXXXX OPCOM, dd-mm-yyyy hh:mm:ss,cc, XXXXXXXXXXXX
Operator _nodename$terminal-name: has been enabled, username USERNAME
```

This message tells you which terminal has been established as an operator terminal and when it was established.

To communicate with you (the operator), users can issue the `REQUEST` command, specifying the `/REPLY` command with the `/TO` qualifier.

If the user issues a `REQUEST/REPLY` command, the request is displayed at the operator terminal in the following format:

```
XXXXXXXXXX OPCOM,dd-mm-yyyy hh:mm:ss,cc XXXXXXXXXXXX
request request-id from user USERNAME __terminal-name:,
"message-text"
```

This message tells you which user sent the message, the time the message was sent, the request identification number assigned to the message, the originating terminal, and the message itself.

If the user issues a REQUEST command, the request is displayed at the operator terminal in a format similar to the one above, but without a request identification number. The message will have the following format:

```
XXXXXXXXXX OPCOM,dd-mm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
request from user USERNAME __terminal-name:, "message-text"
```

When a user issues a REQUEST/REPLY command and you have disabled all operator terminals with the appropriate REPLY/DISABLE commands, OPCOM returns a message to the user indicating that no operator coverage is available, but records all subsequent user's requests in the operator log file. A sample copy of an operator log file that contains typical disk and magnetic tape related requests is included in Section 7.4. For more information on the operator log file, see the *VAX/VMS System Manager's Reference Manual*.

You respond to user requests with one of the following REPLY commands:

- REPLY/ABORT=identification-number "message-text"
- REPLY/PENDING=identification-number "message-text"
- REPLY/TO=identification-number "message-text"
- REPLY/TO=identification-number "SUBSTITUTE device-name"

The /ABORT qualifier indicates that the user request has been canceled. The /PENDING qualifier places the user in a waiting state until the request can be fulfilled or aborted. The /TO qualifier indicates that the user request has been fulfilled. The /TO qualifier, with the word SUBSTITUTE and a device name in place of *message-text*, redirects the mount operation to the specified device. For this qualifier you must load the user's volume on the alternate device and ready the device before you issue the REPLY command. You can abbreviate the word SUBSTITUTE to S and can use upper or lowercase characters. Here is an example:

```
$ REPLY/TO=24 "SUBSTITUTE DMA1"
```

This chapter describes the OPCOM facility and the REPLY command only as they relate to disk and magnetic tape functions handled by operators. For more detailed information on the REPLY command, including a list of all the command qualifiers, see the *VAX/VMS DCL Dictionary*. For more information on OPCOM, see the *VAX/VMS System Manager's Reference Manual*.

---

## 7.2 Handling User Requests for Mounting Volumes

At many installations, operators perform the physical mounting and dismounting of both system and private disk and magnetic tape volumes. At these installations, users do the following:

- Allocate a device of the required type.
- Enter the MOUNT or MOUNT/COMMENT command to issue the mount request and send a message to you specifying the following: the name of the device allocated, the volume requested to be mounted, the physical identification on the outside of the volume, and its location.
- Wait until the system responds with a message indicating that the volume is mounted.

As an operator handling user requests for mounting volumes, you do the following:

- Place the volume on the specified device.
- Perform the necessary start-up procedure. On disk drives, the start-up procedure requires pressing the START or RUN button; on magnetic tape drives, it requires pressing the LOAD button.

If the user intends to write on a magnetic tape volume to be mounted, make sure the magnetic tape contains a write ring before you load the volume on the drive. A volume that does not contain a write ring is write-locked and thus cannot be accessed for write operations.

If the user wants a particular disk to be checked for bad blocks, make sure that the volume has been mounted with the /FOREIGN qualifier. For more information on how to invoke and use the Bad Block Locator Utility (BAD), see the *VAX/VMS Bad Block Locator Utility Reference Manual*.

The following sections describe the operator's function in mounting disk and magnetic tape volumes. For more information on the user's role in mounting volumes, see Chapter 3. For a complete list of all of the MOUNT command qualifiers and parameters, see the *VAX/VMS Mount Utility Reference Manual*.

### 7.2.1 Requests for Mounting Disks and Single Magnetic Tape Volumes

When a user requests a specific disk or magnetic tape to be mounted on a device, the following type of message is displayed on the operator terminal:

```
XXXXXXXXXX OPCOM dd:mmm:yyy:hh:mm:ss:cc XXXXXXXXXXXX
request NUMBER, from user USERNAME
```

For example, a user requesting to mount the volume TEST\_FILES on the device DMA2:, could issue the following command:

```
$ MOUNT DMA2: TEST_FILES/COMMENT="Shelf slot 6B"
```

This command notifies you by displaying the following message at the operator terminal:

```
XXXXXXXXXX OPCOM, 11-JUN-1985 15:47:50.26 XXXXXXXXXXXX
request 5, from user MALCOLM
Please mount volume TEST_FILES in device _DMA2:
Shelf slot 6B
```

The user receives a message indicating that you have received the request:

```
%MOUNT-I-OPRQST, Please mount volume TEST_FILES in device _DMA2:
Shelf slot 6B
```

After you locate the physical volume and place it on the device, the user receives the messages:

```
%MOUNT-I-MOUNTED, TEST_FILES mounted on _DMA2:
%MOUNT-I-RQSTDON, operator request canceled -- mount completed successfully.
```

The first message notifies the user that the volume has been mounted. The second informs the user that the operator has been notified that the original user request is no longer outstanding.

Instead of requesting a specific device for mounting a volume, the user can also make a generic MOUNT request by specifying a device type. To mount the volume CITIES on a magnetic tape drive whose name begins with MT, for example, the user would issue the following command:

```
$ MOUNT MT: CITIES/COMMENT="Slot 12c"
```

If the user has already allocated a drive whose name begins with MT, the VAX/VMS Mount Utility (MOUNT) will request that you mount CITIES on that particular drive. If the user has not allocated a drive whose name begins with MT, the MOUNT utility will allocate the first available TE16, TU45, or TU77 tape drive it finds and issue a request to you to mount CITIES on that drive.

---

### 7.2.2 Requests for Mounting Magnetic Tape Volume Sets

The procedure for mounting a magnetic tape volume set is similar to the procedure for mounting a single magnetic tape volume described in the previous section. When mounting a volume set, however, you should make sure that all the volumes in the set contain write rings if the user intends to write to any one of the volumes in the set. If even one of the volumes in the set does not contain a write ring at mount time, all volumes will be write-locked and thus not be able to be written. You can then load the volumes on the drives that have been allocated and place the drives on line.

The next two sections describe how to handle user requests to mount a magnetic tape volume set when:

- Automatic volume switching is enabled
- Automatic volume switching is disabled

---

#### 7.2.2.1 Mounting Volume Sets with Automatic Switching

The VAX/VMS operating system supports automatic volume recognition (AVR) and automatic volume labeling (AVL) for magnetic tapes. If the magnetic tape volume is mounted with AVR and AVL enabled, then the only task required of the operator is to physically place the next volume in the drive. The system will automatically read the label and then mount the volume. The system will then send a message to the operator terminal informing you that a volume switch has occurred.

The magnetic tape file system switches volumes by issuing pending read or write requests to the next volume in the set. For more information on how automatic volume switching occurs, including examples of processing continuation volumes in a volume set with automatic switching, see Chapter 3.

With automatic volume switching, you can load each successive magnetic tape in the volume set on one of the allocated drives any time before the magnetic tape being processed reaches an end-of-tape position. The magnetic tape file system will mount and/or initialize the next magnetic tape, and then notify you that a switch has occurred.

Note that to take advantage of the automatic volume-switching capability, more than one magnetic tape drive must be allocated to the magnetic tape volume set. Automatic volume switching will be implicitly disabled if the user allocates only one magnetic tape drive for the volume set. Note also that a user can explicitly override automatic volume switching by mounting the volume set with the /NOAUTOMATIC qualifier.

#### 7.2.2.2 Mounting Volume Sets without Automatic Switching

This section describes how to handle requests for mounting continuation volumes in a volume set when automatic volume switching is disabled or when the magnetic tape file system cannot mount a given volume.

When a user is in the process of reading or writing to a magnetic tape and that tape reaches an end-of-tape condition, the system suspends processing and sends a request to the operator to mount the next magnetic tape in the volume set. The user does not see this message and may not realize that another tape is needed to complete the read or write operation.

After you load the continuation volume on the drive specified in the mount request, you mount the volume by issuing the `REPLY/TO` command, specifying the identification number in the mount request. If the volume does not restrict access and it is the correct volume, the magnetic tape file system (MTFS) mounts the volume and reissues pending read or write requests to the continuation volume.

If a read operation was being performed when the volume switching occurred, specifying the volume identifier is optional. The MTFS performs checks on the continuation volume to ensure that it is the correct volume.

Before it mounts a volume for which a volume identifier was not specified, the magnetic tape file system performs the access checks described in Chapter 2. If the volume fails any of these access checks, the magnetic tape file system will not mount the volume.

If a write operation is in progress when a continuation volume is needed, the volume identifier of the continuation volume must be specified in either the mount request or the `REPLY` command. Specifying a volume identifier ensures that the correct volume is mounted on the drive and links the continuation volume to the volume set.

To satisfy a mount request from a magnetic tape file system when a write operation is in progress, one of three qualifiers must be specified with the `REPLY` command:

- `/BLANK_TAPE`
- `/INITIALIZE_TAPE`
- `/TO`

The volume identifier and the format written on the continuation volume determine which qualifier should be used.

For example, assume that the magnetic tape file system issued the following mount request:

```
XXXXXXXXXX OPCOM, 14-JUN-1985 15:23:31.78 XXXXXXXXXXXX  
request 3, from user PLAW  
MOUNT new relative volume 2 (DWOQT2) on MTA1:
```

When the volume identifier written on the continuation volume is the same as that specified in the mount request, the /TO qualifier should be used with the REPLY commands to satisfy the request, as shown in the two examples below.

```
$ REPLY/TO=3  
$ REPLY/TO=3 "DWOQT2"
```

The first REPLY command does not specify the volume identifier; the second does.

If you initialize and mount a volume set in which each volume has a unique accessibility character and you wish to retain that accessibility character, you must not specify the volume identifier. If you do specify the volume identifier, as in the second REPLY command, the accessibility character of the first volume in the set that was mounted will be written to the continuation volume, rather than the character specified at initialization.

When the volume identifier on the continuation volume does not match the one specified in the mount request, the /INITIALIZE\_TAPE qualifier should be specified with the REPLY command to reinitialize and mount the volume with the volume identifier.

The magnetic tape file system will then perform access checks and initialize the volume as if the INITIALIZE command had been specified (see Chapter 3). However, if the volume fails any of the access checks of the magnetic tape file system, the magnetic tape file system cannot override them, and the volume cannot be initialized.

The two examples that follow illustrate uses of the /INITIALIZE\_TAPE qualifier.

```
$ REPLY/INITIALIZE_TAPE=3  
$ REPLY/INITIALIZE_TAPE=3 "DWOQT2"
```

The first example does not specify the volume identifier; the second does.

When a volume is unformatted, the /BLANK\_TAPE qualifier must be specified to initialize the volume. The user privileges VOLPRO and OPER are required to avoid a runaway tape or timeout condition (see Chapter 3). Either of these next REPLY commands is valid.

```
* REPLY/BLANK_TAPE=3
```

```
* REPLY/BLANK_TAPE=3 "DWOQT2"
```

The first command specifies a volume identifier; the second does not.

---

### 7.3 Handling Requests from the Backup Utility

When a backup operation is submitted as a batch job, you receive requests to load the next volume of a save set and to reload a volume in the event of an error. (These requests go to the user when the Backup Utility is invoked interactively.)

---

#### 7.3.1 Writing to a Save Set

If a save operation (writing from files to a save set) requires the loading of an additional volume, you receive messages stating the date and time, a request number, the user name, and the device name, as shown in the following example:

```
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:32.31 XXXXXXXXXXXX
Request 24, from user TOM.
%BACKUP-I-READYWRITE, mount volume 2 on _MTA0: for writing
```

To continue the backup operation, load a scratch volume, ready the device, and type a REPLY/TO command, as shown in the following example:

```
* REPLY/TO=24
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:34.14 XXXXXXXXXXXX
Request 24 completed by operator OPA0.
```

You can also abort the backup operation by typing a REPLY/ABORT command, as shown in the following example:

```
* REPLY/ABORT=24
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:34.14 XXXXXXXXXXXX
Request 24 aborted by operator OPA0.
```

### 7.3.2 Reading from a Save Set

If a restore operation (reading from a save set to files) requires the loading of an additional volume, you receive messages stating the date and time, a request number, the user name, and the device name, as shown in the following example:

```
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:32.31 XXXXXXXXXXXX
Request 24, from user TOM.
%BACKUP-I-READYREAD, mount volume 2 on _MTA0: for reading
```

To continue the restore operation, place the next volume of the save set on the drive, ready the device, and type a REPLY/TO command, as shown in the following example:

```
$ REPLY/TO=24
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:34.14 XXXXXXXXXXXX
Request 24 completed by operator OPA0.
```

You can also abort the restore operation by typing a REPLY/ABORT command, as shown in the following example:

```
$ REPLY/ABORT=24
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:34.14 XXXXXXXXXXXX
Request 24 aborted by operator OPA0.
```

### 7.3.3 Recovering from an Error

On certain errors, you receive messages stating the date and time, a request number, the user name, the device name, and reply options. On read errors from the save set, the options are usually CONTINUE and QUIT. On write errors to the save set, the options are usually RESTART and QUIT. You should take one of the following actions:

- REPLY/TO "CONTINUE" — If you can fix the problem and are given the CONTINUE option, ready the device, and issue a REPLY/TO command specifying the word CONTINUE as text.
- REPLY/TO "RESTART" — If you can fix the problem and are given the RESTART option, load (if necessary) the volume, ready the device, and issue a REPLY/TO command specifying the word RESTART as text.
- REPLY/TO "QUIT" — If you cannot fix the problem, issue a REPLY/TO command specifying the word QUIT as text.

The text in the REPLY/TO command can be uppercase or lowercase, and can be abbreviated down to the first character. See the *VAX/VMS Error Log Utility Reference Manual* for additional information on errors and recovery procedures.

Assume, for example, that you receive the following messages during a restore operation (the save set is being read):

```
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:30.58 XXXXXXXXXXXX
Message from user RESJOB.
%BACKUP-E-FATALERR, fatal error on MT: []SAVE.;
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:30.89 XXXXXXXXXXXX
Message from user RESJOB.
-SYSTEM-F-MEDOFI, medium is offline
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:31.05 XXXXXXXXXXXX
Request 24, from user RESJOB.
%BACKUP-I-SPECIFY, specify option (QUIT or CONTINUE)
```

You check the drive and find that it simply lost its vacuum. Remedy the situation by readying the tape drive and issuing the following command:

```
$ REPLY/TO=24 "CONTINUE"
CONTINUE
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:33.41 XXXXXXXXXXXX
Request 24 completed by operator OPA0.
```

If you find the drive inoperable, you issue the QUIT reply:

```
$ REPLY/TO=24 "QUIT"
QUIT
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:33.41 XXXXXXXXXXXX
Request 24 completed by operator OPA0.
```

If the error occurs during a save operation (the save tape is being written), you have a choice of QUIT or RESTART:

```
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:30.58 XXXXXXXXXXXX
Message from user RESJOB.
%BACKUP-E-FATALERR, fatal error on MT: []SAVE.;
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:30.89 XXXXXXXXXXXX
Message from user RESJOB.
-SYSTEM-F-MEDOFI, medium is offline
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:31.05 XXXXXXXXXXXX
Request 24, from user RESJOB.
%BACKUP-I-SPECIFY, specify option (QUIT or RESTART)
(You ready the magnetic tape drive.)
```

```
$ REPLY/TO=24 "RESTART"
RESTART
XXXXXXXXXX OPCOM, 6-JUN-1985 17:02:33.35 XXXXXXXXXXXX
Request 24 completed by operator OPA0.
```

The RESTART option permits you to restart a multivolume backup operation at the beginning of the current volume. If you specify the QUIT option, you must restart the backup operation completely.

## 7.4 Handling Mount Verification Tasks

The mount verification feature of Files-11 disk handling generally protects users in situations where a mounted disk has gone off line and returned on line or in some other way has become unreachable and then restored. Mount verification is enabled by default with the /MOUNT\_VERIFICATION qualifier when the disk is mounted. To disable mount verification, the user must have specified /NOMOUNT\_VERIFICATION when mounting the disk. Note that this feature does not apply to foreign disks or to magnetic tapes.

Mount verification sends messages to OPCOM. Because there are cases where mount verification messages are needed at the operator console and OPCOM might not be able to provide them, mount verification also sends special messages with the prefix %SYSTEM-I-MOUNTVER to the system console only, that is, to OPA0:. For example, if the system disk undergoes a mount verification or if the OPCOM process is not present on a system, you would at least receive the messages with the %SYSTEM-I-MOUNTVER prefix. Under normal circumstances, both messages are received at OPA0:, with the %SYSTEM-I-MOUNTVER message arriving first.

### 7.4.1 Mount Verification for Offline Devices

If a mounted disk volume goes off line while mount verification is enabled, you can follow the steps in the procedure below to resume operations.

#### Procedure

When a device is taken off line, the steps in mount verification are as follows:

- 1 Because of a hardware or user error, a disk volume is taken off line (for example, it might be spun down). Most disk drives generate a special interrupt when the volume comes back on line, which causes the software to mark the volume as invalid. However, some disk drives (such as the RX01, RX02, RL02, and TU58) do not generate online attention interrupts. For these devices, an offline condition is only detected when an I/O operation is initiated for the drive.
- 2 An I/O operation fails with a "medium offline" or "volume invalid" status. The software marks the volume to indicate that it is undergoing mount verification, and all I/O operations to the disk are stalled.
- 3 OPCOM issues a message to the operators enabled for DISKS and DEVICES to announce the disk's unavailability, as follows:

```

XXXXXXXXXXXX OPCOM, dd-mmm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
Device device-name is offline.
Mount verification in progress.
  
```

- 4 You may simply choose to take the disk out of the offline and verification-pending state by shutting down mount verification with one of the techniques described in Sections 7.3.3 and 7.3.4. These techniques cause the pending and future I/Os to the volume to fail.
- 5 Otherwise, you take corrective action. Perhaps the drive can be brought back on line. If the disk drive is faulty, but another functioning drive is available on the same controller, you can move the disk to the functioning drive and swap the unit select plugs.
- 6 The disk comes back on line, which is detected by the mount verification software that polls the disk drive.
- 7 The system verifies that the disk's home block matches the one in the database of mounted volumes, thus confirming that this is the same disk as the one previously mounted.
- 8 If the drive does not contain the correct volume, OPCOM issues the following message:

```
XXXXXXXXXXXX OPCOM, dd-mmm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
Device device-name contains the wrong volume.
Mount verification in progress.
```

- 9 After the mount verification succeeds, the disk is marked as valid. OPCOM issues the following message:

```
XXXXXXXXXXXX OPCOM, dd-mmm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
Mount verification completed for device device-name
```

At this point I/O operations to the disk are allowed to proceed.

### Example

```
XXXXXXXXXXXX OPCOM, 15-JUN-1985 11:54:54.12 XXXXXXXXXXXX
Device DMA0: is offline.
Mount verification in progress.
XXXXXXXXXXXX OPCOM, 15-JUN-1985 11:57:34.22 XXXXXXXXXXXX
Mount verification completed for device DMA0:
```

The message from OPCOM informs you that device DMA0: has gone off line and mount verification has been initiated. You find that the drive was accidentally spun down and successfully spin it back up. The next message indicates that mount verification is satisfied that the same volume is on the drive (which it has found is on line again), and all I/Os to the volume resume.

Note that if a device continues to go off line because of hardware problems, you should consult your DIGITAL Field Service Representative.

### 7.4.2 Mount Verification for Write-Locked Devices

If for some reason a mounted disk volume becomes write-locked while mount verification is enabled, you can follow the steps in the procedure below to resume operations.

#### Procedure

- 1 A Files-11 disk volume is mounted for writing (the /WRITE qualifier is the default). Through some hardware or user error, the disk becomes write-locked.
- 2 The software discovers that the disk is write-locked (typically an I/O fails with a write-lock error). The disk is marked that verification is in progress. OPCOM issues a message to the operators enabled for DISKS and DEVICES to announce the disk's unavailability, as follows:

```
XXXXXXXXXXXX OPCOM, dd-mmm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
Device device-name has been write-locked.
Mount verification in progress.
```

- 3 You may simply choose to take the disk out of the verification pending state by shutting down mount verification with one of the techniques described in Sections 7.3.3 and 7.3.4. These techniques cause the pending and future I/Os to the volume to fail.
- 4 Otherwise, you take corrective action. Perhaps the drive can simply be write enabled by toggling the drive's hardware write-lock switch. If the disk drive is faulty, but another functioning drive is available on the same controller, you can move the disk to the functioning drive and swap the unit select plugs. (Note that switching to another drive will cause the volume to undergo offline mount verification. Once that completes, the write-lock mount verification continues.)
- 5 The mount verification software that polls the disk drive determines that the volume is in a writable state. At this point, I/O operations to the disk are allowed to proceed. However, OPCOM does not issue a message indicating that the write-lock mount verification has completed.

#### Example

```
XXXXXXXXXXXX OPCOM, 29-JUN-1985 15:28:54.07 XXXXXXXXXXXX
Device DMA1: has been write-locked.
Mount verification in progress.
```

The OPCOM message informs you that device DMA1: is write-locked. You toggle the write-lock switch on the drive to eliminate the write lock. I/O operations to the disk resume, with no further messages.

### 7.4.3 Canceling Mount Verification

If you fail to either bring a system disk back on line or to write-enable it as appropriate while mount verification is enabled, the whole system can hang, because normal system operation depends on performing I/O to the system disk. This problem extends to other public disks that contain secondary paging and swap files, the authorization database, and so forth. Failure to bring such a disk back on line is generally fatal to the system.

In addition, obscure interdependencies within VAX/VMS may cause the system to suspend activity in rare circumstances — even when a nonsystem disk goes off line. If this type of problem occurs, you can clear it by using either the first or second technique described below for canceling mount verification requests.

Normally, you can cancel a mount verification request in one of three ways:

- 1 Allow the mount verification in progress to continue for the number of seconds defined by the system parameter MVTIMEOUT. When the time expires, the system automatically cancels the pending mount verification. Note that a mount verification initiated by a write-lock will not time out.
- 2 Invoke a special canceling routine from the console terminal.
- 3 Dismount the volume with the DCL command DISMOUNT from a process that is not hung.

### 7.4.4 Dismounting a Volume to Abort Mount Verification

In some cases, a user can abort mount verification by dismounting the volume in question. This only works if it is possible for the user to issue the DCL command DISMOUNT for the volume (for example, if the system file ACP is not hung).

#### Procedure

- 1 Log in at another terminal or use any logged-in terminal that has sufficient privilege to dismount the volume.
- 2 Enter the DCL command DISMOUNT/ABORT for the volume
- 3 When you cancel a pending mount verification by dismounting the volume, OPCOM issues the following message:

```
XXXXXXXXXX OPCOM, dd-mm-yyyy hh:mm:ss.cc XXXXXXXXXXXX
Mount verification aborted for device device-name
```

If you do not have access to the volume, you will receive an error message. You can try again if you can find an appropriate process to use. If your process hangs, it is the system file ACP that is hung (first condition above was not met), and you cannot use this technique to cancel mount verification.

- 4 Once the cancellation succeeds, remove the volume from the drive.

For more information on Mount Verification, see the description in *VAX/VMS System Manager's Reference Manual*.



# A

## VAX/VMS Disk Files and Volumes

### A.1 Files-11 Disk Structure

The VAX/VMS system recognizes two disk file structures: Files-11 On-Disk Structure Level 1 and Files-11 On-Disk Structure Level 2. Files-11 On-Disk Structure Level 2 is the default disk structure of the VAX/VMS system, and Files-11 On-Disk Structure Level 1 is a structure used by DIGITAL's RSX-11M, RSX-11D, RSX-11M-PLUS, and IAS operating systems.

Nine files control the structure of a Files-11 On-Disk Structure Level 2 volume. Only five of these files are used for a Files-11 On-Disk Structure Level 1 volume. Table A-1 identifies all nine files, which are referred to as reserved files, and indicates to which Files-11 On-Disk Structure Level they pertain.

**Table A-1 VAX/VMS Reserved Files**

Reserved File	File Name	Structure Level 1	Structure Level 2
Index file	INDEXF.SYS;1	X	X
Storage bit map file	BITMAP.SYS;1	X	X
Bad block file	BADBLK.SYS;1	X	X
Master file directory	000000.DIR;1	X	X
Core image file	CORIMG.SYS;1	X	X
Volume set list file	VOLSET.SYS;1		X
Continuation file	CONTIN.SYS;1		X
Backup log file	BACKUP.SYS;1		X
Pending bad block	BADLOG.SYS;1		X

All the files listed in Table A-1 are listed in the master file directory (MFD), [000000].

---

### A.1.1 Index File

Every Files-11 volume has an index file, which is created when the volume is initialized. This index file identifies the volume to the operating system as a Files-11 structure and contains the access data for all files on the volume. The index file, which is listed in the master file directory as INDEXF.SYS;1, contains the following information:

- **Bootstrap block** — The volume's bootstrap block is virtual block number 1 of the index file. If the volume is a system volume, this block contains a bootstrap program that loads the operating system into memory. If the volume is not a system volume, this block contains a program that displays a message that the volume is not the system device but a device that contains users' files only.
- **Home block** — The home block establishes the specific identity of the volume, providing such information as the volume name and protection, the maximum number of files allowed on the volume, and the volume ownership information. The home block is virtual block number 2 of the index file.
- **Backup home block** — The backup home block is a copy of the home block. It permits the volume to be used even if the primary home block is destroyed.
- **Backup index file header** — The backup index file header permits recovery of data on the volume if the index file header becomes damaged.
- **Index file bit map** — The index file bit map controls the allocation of file headers and thus the number of files on the volume. The bit map contains a bit for each file header that is allowed on the volume. If the value of a bit for a given file header is 0, a file can be created with this file header. If the value is 1, the file header is already in use.
- **File headers** — The largest part of the index file is made up of file headers. Each file on the volume has a file header, which describes such properties of the file as file ownership, creation date and time, file protection, and location of the data in the file. The file header points to all the information needed for gaining access to the file.

---

### A.1.2 Storage Bit Map File

The storage bit map file controls the available space on a volume; this file is listed in the master file directory as BITMAP.SYS;1. It contains a storage control block, which consists of summary information intended to optimize the Files-11 space allocation, and the bit map itself, which lists the availability of individual blocks.

---

**A.1.3 Bad Block File**

The bad block file, which is listed in the master file directory as `BADBLK.SYS;1`, contains all the bad blocks on the volume. The system detects bad disk blocks dynamically and prevents their reuse once the files to which they are allocated have been deleted.

---

**A.1.4 Master File Directory**

The master file directory (MFD) itself is listed in the MFD as `000000.DIR;1`. The MFD, which is the root of the volume's directory structure, lists the reserved files that control the volume structure and may list both users' files and users' file directories. Usually, however, the MFD is used to list the reserved files and users' file directories; users seldom enter files in the MFD, even on private volumes. In fact, on a private volume, it is most convenient for a user to create a directory that has the same name as the user's default directory on a system disk. For an explanation of users' file directories and file specifications, see the *VAX/VMS DCL Concepts Manual*.

When the VAX/VMS Backup Utility (BACKUP) creates sequential-disk save sets, it stores the save-set file in the MFD. (See Section 6.4.3.2 for a discussion of sequential-disk save sets.)

---

**A.1.5 Core Image File**

The core image file is listed in the MFD as `CORIMG.SYS;1`. It is not supported by VAX/VMS.

---

**A.1.6 Volume Set List File**

The volume set list file is listed in the MFD as `VOLSET.SYS;1`. This file is used only on relative volume 1 of a volume set. The file contains a list of the labels of all the volumes in the set and the name of the volume set.

---

**A.1.7 Continuation File**

The continuation file is listed in the MFD as `CONTIN.SYS;1`. This file is used as the extension file identifier when a file crosses from one volume to another volume of a loosely coupled volume set. This file is used for all but the first volume of a sequential-disk save set (see Section 6.4.3.2 for a discussion of sequential-disk save sets).

---

### A.1.8 Backup Log File

The backup log file is listed in the MFD as BACKUP.SYS;1. This file is reserved for future use.

---

### A.1.9 Pending Bad Block Log File

The pending bad block log file is listed in the MFD as BADLOG.SYS;1. This file contains a list of suspected bad blocks on the volume that are not listed in the bad block file.

---

### A.1.10 Files—11 On-Disk Structure Level 1 Versus Structure Level 2

Files-11 On-Disk Structure Level 2, a compatible superset of Structure Level 1, is the preferred disk structure on VAX/VMS for reasons of performance and reliability. At volume initialization time (see the INITIALIZE command in the *VAX/VMS DCL Dictionary*), Structure Level 2 is the default. Structure Level 1 should be specified only for volumes that must be transportable to RSX-11M, RSX-11D, RSX-11M-PLUS, and IAS systems, as these systems support only that structure level. Additionally, you may be required to handle Structure Level 1 volumes transported to VAX/VMS from one of the above systems.

Where Structure Level 1 volumes are in use on the system, you should bear in mind the following limitations on them:

- Directories — No hierarchies of directories and subdirectories, and no ordering of directory entries (that is, the file names) in any way (RSX-11M, RSX-11D, RSX-11M-PLUS, and IAS systems do not support subdirectories and alphabetical directory entries)
- Disk quotas — Not supported
- Multivolume files and volume sets — Not supported
- Placement control — Not supported
- Caches — No caching of file header blocks, file identification slots, or extent entries
- System disk — Cannot be a Structure Level 1 volume
- VAXcluster access — Local access only; cannot be shared across a cluster
- Clustered allocation — Not supported
- Backup home block — Not supported

## VAX/VMS Disk Files and Volumes

- Protection code E — Means extend for RSX-11M but is ignored by VAX/VMS
- File versions — Limited to 32,767; version limits are not supported
- Enhanced protection features (for example, access control lists) are not supported

Future enhancements to VAX/VMS will be based primarily on Structure Level 2, so that further restrictions on Structure Level 1 volumes may be incurred.



# B

## VAX/VMS ANSI-Labeled Magnetic Tape

This appendix briefly describes ANSI labels, data, and record formats supported by VAX/VMS. It also describes support for the various ANSI labels on VAX/VMS. For a complete description of these labels, please refer to the ANSI standard (American National Standard X3.27-1978). Use this standard to write VAX/VMS standard tapes.

### B.1 Logical Format of ANSI-Labeled Volumes

The format of VAX/VMS ANSI-labeled magnetic tape volumes is based on Level 3 of the ANSI standard for magnetic tape labels and file structure for information interchange. This standard specifies the format, content, and sequence of volume labels, file labels, and file structures. According to this standard, volumes are written and read on 9-track magnetic tape drives only. The contents of labels must conform to prescribed data and record formats. All labels must consist of ASCII "a" characters.

The VAX/VMS ANSI-labeled format allows you to write binary data in the file sections (see Figure B-1) of files. However, if you plan to use such files for information interchange across systems, make sure that the recipient system can read the binary data.

### B.2 VAX/VMS Magnetic Tape Ancillary Control Process (MTAACP)

The VAX/VMS magnetic tape ancillary control process (MTAACP) is the internal software process of the operating system that interprets the logical format of VAX/VMS ANSI-labeled volumes. Transparent to your process, the MTAACP process reads, writes, and interprets VAX/VMS ANSI labels before passing this information to VAX Record Management Services (RMS) and Queue I/O (\$QIO) system services. These services in turn read, write, and interpret the record format of the data written in the file section.

---

### **B.3 Basic Components of the VAX/VMS ANSI-Labeled Format**

The format of VAX/VMS ANSI-labeled magnetic tape consists of four basic components:

- Beginning-of-tape (BOT) and end-of-tape (EOT) markers
- tape marks
- file sections
- volume, header, and trailer labels.

Figure B-1 displays the arrangement and function of these components.

---

#### **B.3.1 Beginning-of-Tape and End-of-Tape Markers**

Every volume has beginning-of-tape (BOT) and end-of-tape (EOT) markers. These markers are pieces of photorefective tape that delimit the writable area on a volume. ANSI standards require that a minimum of 14 feet to a maximum of 18 feet of magnetic tape precede the BOT marker; a minimum of 25 feet to a maximum of 30 feet of magnetic tape, of which 10 feet must be writable, must follow the EOT marker. The EOT marker indicates the start of the end of the writable area of the tape, rather than the physical end of the tape. Therefore, data and labels can be written after the EOT marker.

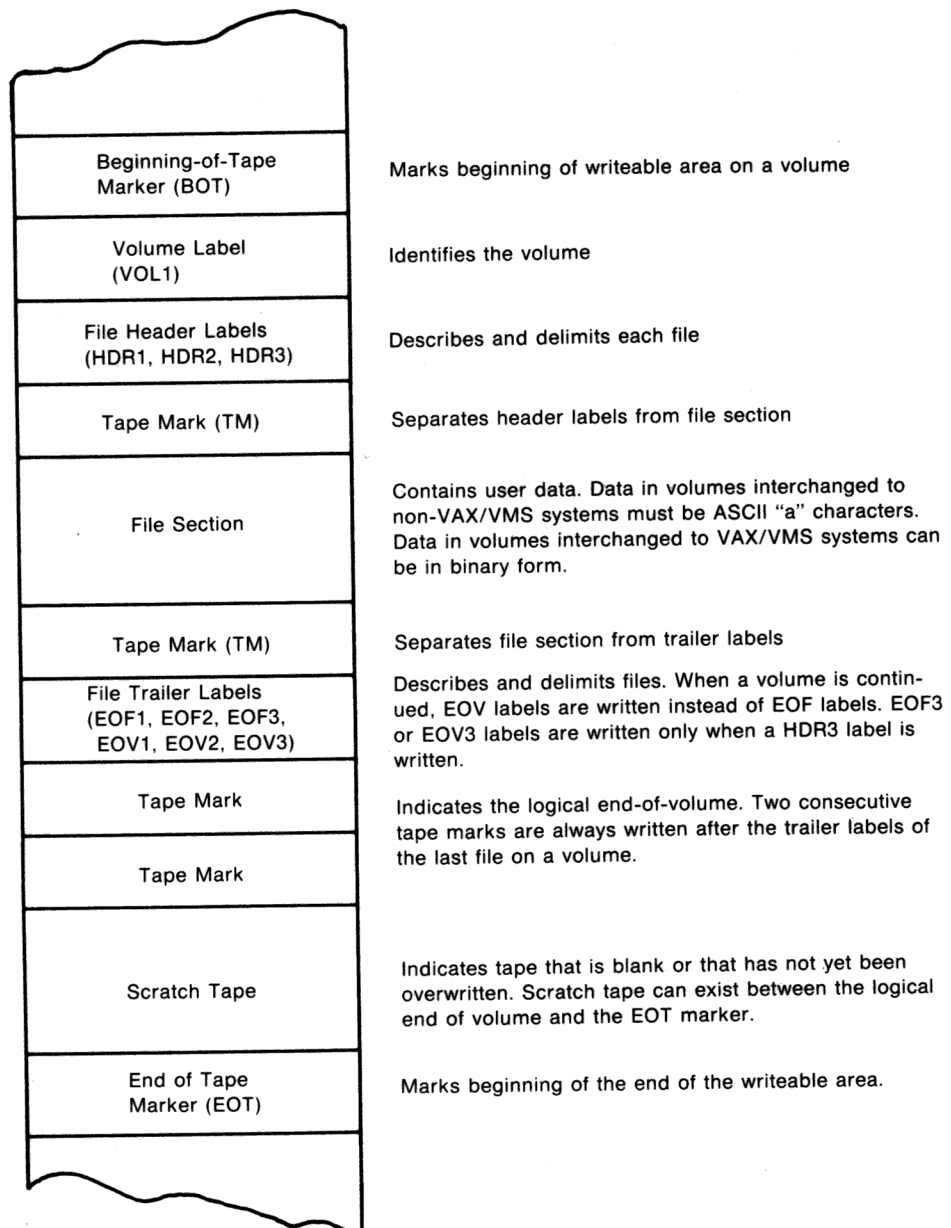
---

#### **B.3.2 Tape Marks**

Tape marks separate the file labels from the file sections, separate one file from another, and denote the logical end-of-volume. On the basis of the number and relative placement of tape marks written on a volume, VAX/VMS determines whether a tape mark delimits a label, a file, or a volume.

Tape marks are written both singly and in pairs. Single tape marks separate either a file section from the header and trailer labels or one file from another. When written after a set of header labels, a single tape mark signals the beginning of a file section. When written before a set of trailer labels, a single tape mark indicates the end of a file section. When written after a trailer label set, a single tape mark separates one file from another.

Double tape marks indicate that either an empty file section exists or that the logical end-of-volume has been reached. A VAX/VMS system creates an empty file when a volume is initialized.

**Figure B-1 Basic Layout of a VAX/VMS ANSI-Labeled Volume**

ZK-981-82

---

### B.3.3 Labels

Labels identify, describe, and control access to volumes and their files. VAX/VMS ANSI-labeled format supports volume, header, and trailer labels. The volume labels are the first labels written on a volume. They identify the volume and the volume owner and specify access protection. Header and trailer labels are sets of labels that identify, describe, and delimit files. Header labels precede files; trailer labels follow files.

Table B-1 lists the labels supported by VAX/VMS. All other ANSI labels are ignored by VAX/VMS on input.

Although each type of label uses a different format to organize its contents, all labels conforming to Version 3 of the ANSI standard must consist of ASCII "a" characters. Some labels contain reserved fields designed for future system use or future ANSI standardization. Reserved fields also must consist of ASCII "a" characters; however, VAX/VMS ignores these characters on input.

---

## B.4 Volume and File Configurations

VAX/VMS ANSI-labeled volumes support four file/volume configurations:

- single file/single volume
- single file/multivolume
- multifile/single volume
- multifile/multivolume.

All these configurations conform to the following guidelines:

- The file sequence number field allows as many as 9999 file sections for one file. In effect, the file length is unlimited.
- Only one file section of a given file is written on a volume.
- When multiple sections exist for one file, each file section is written to a separate volume in the volume set. The file section numbers of each section are written consecutively in ascending order (section n+1 is written immediately following section n); file sections of other files are not interspersed.

Each of the file/volume configurations is illustrated in the subsections that follow.

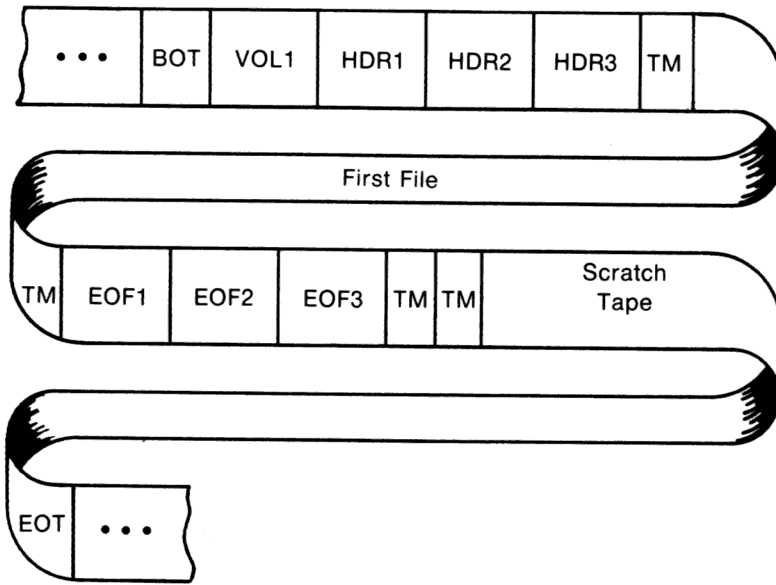
**Table B-1 Labels and Components Supported by VAX/VMS**

Symbol	Meaning
BOT	Beginning-of-tape marker
EOF1	First end-of-file label
EOF2	Second end-of-file label
EOF3	Third end-of-file label
EOF4	Fourth end-of-file label
EOT	End-of-tape marker label
EOV1	First end-of-volume label
EOV2	Second end-of-volume label
EOV3	Third end-of-volume label
EOV4	Fourth end-of-volume label
HDR1	First header label
HDR2	Second header label
HDR3	Third header label
HDR4	Fourth header label
VOL1	First volume label
VOL2	Second volume label
TM	Tape mark
TM TM	Double tape mark indicates an empty file section or the logical end-of-volume

### B.4.1 Single File/Single Volume Configuration

A single file/single volume configuration consists of one file on one volume. The components of the ANSI-labeled format for this configuration are illustrated in Figure B-2.

**Figure B-2 Single File/Single Volume Configuration**

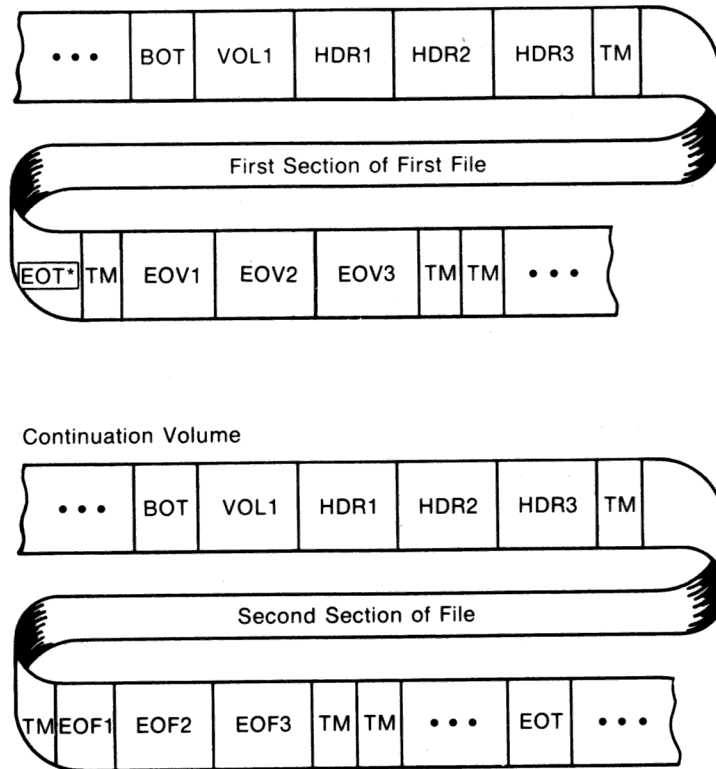


ZK-346-81

### B.4.2 Single File/Multivolume Configuration

A single file/multivolume configuration consists of one file that spans two or more volumes in a volume set. Figure B-3 illustrates the components of the ANSI-labeled format for this configuration.

**Figure B-3 Single File/Multivolume Configuration**



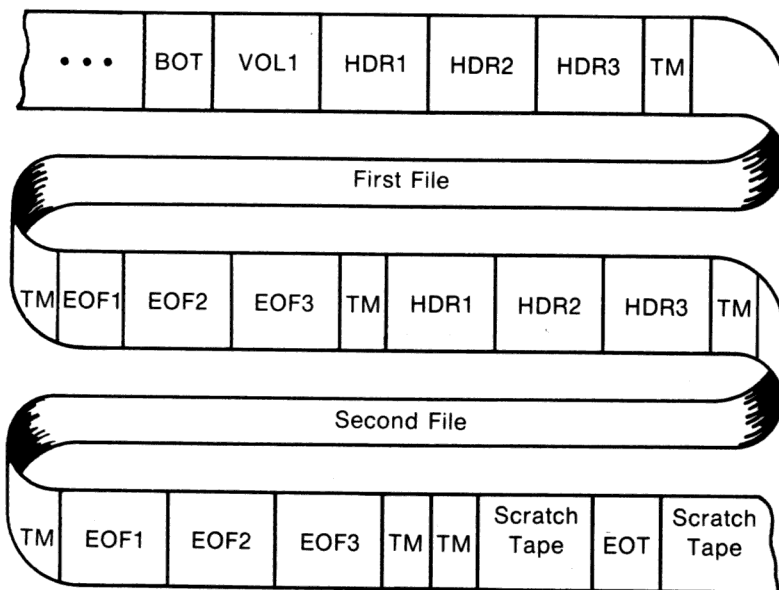
\*When the driver encounters an EOT marker during a write operation, the MTAACP writes the appropriate trailer labels and performs a volume switch, if necessary.

ZK-347-81

### B.4.3 Multifile/Single Volume Configuration

A multifile/single volume configuration consists of two or more files on a single volume. It is the most common file and volume configuration. Figure B-4 illustrates the components of the ANSI-labeled format for this configuration.

**Figure B-4 Multifile/Single Volume Configuration**

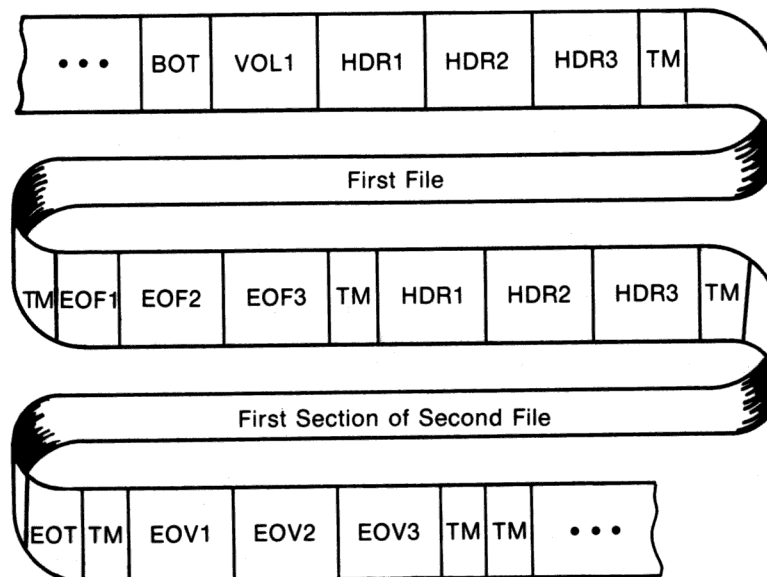


ZK-345-81

#### B.4.4 Multifile/Multivolume Configuration

A multifile/multivolume configuration consists of two or more files that span two or more volumes in the same volume set. Figure B-5 illustrates the components of the ANSI-labeled format for this configuration.

**Figure B-5 Multifile/Multivolume Configuration**



ZK-348-81

### B.5 VOLUME LABELS

The sections that follow describe the first volume (VOL1) and second volume (VOL2) labels.

#### B.5.1 VOL1 Label

The 80-character volume label (VOL1) is the first label written on a VAX/VMS ANSI-labeled volume. It defines the label type, name, and owner of the volume. Although there are many fields in a VOL1 label, this section describes only those fields that you can access or that can inhibit access to a volume and its files on VAX/VMS.

---

#### **B.5.1.1 Volume Identifier Field**

The volume identifier field is a 6-character field that contains the name of the volume. You specify the volume identifier in the command string when you initialize or mount a volume (see Chapter 4). The volume identifier consists of six ASCII "a" characters. Lowercase characters are not in the "a" set, but if you specify them, VAX/VMS will change them to uppercase. If you specify fewer than six characters, VAX/VMS pads the field by right-justifying the field with the ASCII space character.

---

#### **B.5.1.2 Accessibility Field**

The accessibility field is a 1-character field that allows an installation to control access to a volume. See Section 2.1.3 for a description of accessibility support.

---

#### **B.5.1.3 Implementation Identifier Field**

The implementation identifier field contains the identifier of the implementation that creates the magnetic tape. This field controls how certain implementation-specific fields and volume labels are interpreted. The magnetic tape file system's implementation identifier is DECFILEII A.

---

#### **B.5.1.4 Owner Identifier Field**

The owner identifier field is available to the user. This field does not affect the checking of a user's access to a volume, except as noted in Chapter 2 of this guide.

---

### **B.5.2 VOL2 Label**

In addition to the first volume (VOL1) label described above, VAX/VMS also provides a second volume (VOL2) label.

The next section describes the volume-owner field of the VOL2 label.

---

**B.5.2.1 Volume-Owner Field**

The volume-owner field contains the VAX/VMS protection information that has been written on the magnetic tape. A second volume label is written only if a VAX/VMS protection scheme had been specified on either the MOUNT or INITIALIZE command.

The volume-owner field also contains a value that incorporates the user identification code (UIC) with the VAX/VMS protection code specified for a volume. By default, VAX/VMS does not write a UIC to this field, thus allowing all users READ and WRITE access. Note, however, that EXECUTE and DELETE access are not applicable to magnetic tape volumes. Also note that regardless of the protection code that you specify, both SYSTEM users and the volume owner always have READ and WRITE access to a volume. The contents of the volume-owner field depends on the VAX/VMS protection code that you specify.

---

**B.6 Header Labels**

VAX/VMS supports four file header labels: HDR1, HDR2, HDR3, and HDR4. The HDR3 and HDR4 labels are optional. The following sections describe and illustrate each file header label.

---

**B.6.1 HDR1**

Every file on a volume has a HDR1 label, which identifies and describes the file by supplying the VAX/VMS MTAACP with the following information:

- File identifier
- File-set identifier
- File section number
- File sequence number
- Generation and generation version numbers
- File creation and expiration dates
- Accessibility code
- Implementation identifier

### B.6.1.1 File Identifier Field

The file identifier field contains the first 17 characters of the file name that you specify. The remainder of the file name is written into the HDR4 label, provided that this label is allowed. If no HDR4 label is supported, a file name longer than 17 characters will be truncated. You may use either an ANSI file name or a VAX/VMS file specification of the following format:

`filename.type;version`

VAX/VMS file specifications are a subset of ANSI file names. However, ANSI file names are valid only for magnetic tape volumes; VAX/VMS file specifications are valid for disk and tape volumes. Both types of file specifiers are compatible with compatibility mode.

A VAX/VMS file specification consists of a file name, a file type, and an optional version number. Valid file names contain a maximum of 39 characters. Valid file types consist of a period followed by a maximum of 39 characters. The semicolon separates the version number from the file type.

Except for wildcard characters, only the characters: A through Z, 0 through 9, and the special characters ampersand (&), hyphen (-), and dollar sign (\$) are valid for VAX/VMS file names and types. The period and semicolon are the only valid special characters, and they are always separators.

ANSI file names do not have a file type field. An ANSI file name consists of a 17-character name string, a period, a semicolon, and an optional version number. You can specify a name string consisting of a maximum of 17 ASCII "a" characters, but you must enclose the string in a pair of quotation marks (as in, for example, "file name"). When you specify fewer than 17 characters, the string is padded on the right with spaces to the 17-character maximum size. If you specify a file name that has trailing spaces, VAX/VMS truncates them when the file name is returned. In addition, the space-padded field prevents you from specifying a unique file name that consists of spaces.

Although you can specify longer file names — up to 79 characters — only the first 17 characters of the file name will be used in interchange.

The quotation mark character requires special treatment because it is both the file name delimiter and a valid ASCII "a" character that can itself be embedded in the name string. You must specify two quotation marks for each one that you want VAX/VMS to interpret. The additional quotation mark informs VAX/VMS that one of the quotation marks is part of the name string, rather than a delimiter.

Embedded spaces also are valid characters, but embedded tabs are not. Lowercase characters are not in the ASCII "a" character set but if you specify them, VAX/VMS converts them to uppercase characters.

If you do not specify a file type or version number on input, VAX/VMS will supply a period (the default file type) and a semicolon (the default version number). However, the period and semicolon will not be written to this field on the tape.

Although VAX/VMS considers version numbers for ANSI and VAX/VMS file names to be part of the file name specification, the version number of a file is not written to the file identifier field, but is mapped to the generation number and generation version number fields as described in Section B.6.1.4.

Examples below display ANSI file names. The input is the format that you specify. The output shown displays the VAX/VMS format returned to your process and the format written to the label. The number sign (#) in the examples indicates a space character. In the last example, a VAX/VMS file name is enclosed in quotation marks, like an ANSI file name, on input. However, the VAX/VMS operating system returns the file name to the process as a VAX/VMS file name, rather than an ANSI file name. Therefore, when you enclose a valid VAX/VMS file name in quotation marks on input, VAX/VMS parses the file name as a VAX/VMS file name.

Input	Output to User Process	Output to HDR1 Label
"AB2&D"FGHI*k4"#-M";2	"AB2&D"FGHI*K4"#-M";2	AB2&D"FGHI*K4"#-M
"#####"	".";	#####
#####;	#####;	#####
"DWDEVOP.DAT"	DWDEVOP.DAT;	DWDEVOP.DAT####
"VMS_LONG_FILENAME.LONG_FILETYPE"	VMS_LONG_FILENAME.LONG_FILETYPE	VMS_LONG_FILENAME

#### B.6.1.2 File-Set Identifier Field

The 6-character file-set identifier field denotes all files that belong to the same volume set. The file-set identifier for any file within a given volume set should always be the same as the file-set identifier of the first file on the first volume that you mount. For VAX/VMS, the file-set identifier is the same as the volume identifier of the first volume that you mount.

#### B.6.1.3 File Section Number and File Sequence Number Fields

The file section number is a 4-character field that specifies the number of the file section.

The file sequence number is a 4-character field that specifies the number of the file in a file set.

#### B.6.1.4 Generation Number and Generation-Version Number Fields

The generation number (a decimal number from 0001 to 9999) and generation-version number (a 2-digit decimal number) fields store the file version number that is specified on input and written by VAX/VMS on output. The VAX/VMS operating system does not increment the version number of a file, even when the version of the specified file already exists on the volume. Therefore, if the file that you specify has the same file name and version number as an existing file, you will have at least two files with the same version number on the same volume set.

On input, VAX/VMS computes the version number by using this calculation:

```
version number =
[(generation number - 1)100] + generation-version number + 1
```

Version numbers larger than 32767 are divided by 32768; the integer remainder becomes the version number.

On output, the generation number is derived from the version number with this calculation:

```
generation number = [(version number - 1)/100] + 1
```

If there is a remainder after the version number is divided by 100, the remainder becomes the generation-version number. It is not added to 1 to form the generation number.

#### B.6.1.5 Creation Date and Expiration Date Fields

The creation date field contains the date the file is created. The expiration date field contains the date the file expires. VAX/VMS interprets the expiration date of the first file on a volume as the date that both the file and the volume expire. VAX/VMS stores the creation and expiration dates in the Julian format. This 6-character format (#YYDDD) consists of a space (#), the year, and the day. Only dates are relevant for these fields; time is always returned as 00:00:00:00.

By default, VAX/VMS writes the current date to both the creation and expiration date fields when you create a file. Because the expiration date is the same as the creation date, the file expires on creation and you can overwrite it immediately. If the expiration date is a date that is later than the creation date and if the files that you want to overwrite have not expired, you must specify the /OVERRIDE=EXPIRATION qualifier with the INITIALIZE or MOUNT command.

To write dates other than the VAX/VMS defaults in the date fields in this label, specify the creation date field (CDT) and the expiration date field (EDT) of the RMS date and time extended attribute block (XABDAT).

When you do not specify a creation date, RMS defaults the current date to the creation date field.

To specify a zero creation date, you must specify a year before 1900. If you specify a binary zero in the date field, the system will write the current date to the field.

For details on the XABDAT, see the *VAX Record Management Services Reference Manual*.

---

#### **B.6.1.6 Accessibility Field**

The contents of this field are described previously, in the volume label use of accessibility.

---

#### **B.6.1.7 Implementation Identifier Field**

The implementation identifier field specifies, using ASCII "a" characters, an identification of the implementation that recorded the Volume Header Label Set.

---

### **B.6.2 HDR2 LABEL**

The HDR2 label describes the record format, maximum record size, and maximum block size of a file.

---

#### **B.6.2.1 Record Format Field**

The record format field specifies the type of record format the file contains. The VAX/VMS operating system supports two record formats: fixed length (F) and variable length (D). When files contain record formats that VAX/VMS does not support, VAX/VMS cannot interpret the formats and classifies them as undefined.

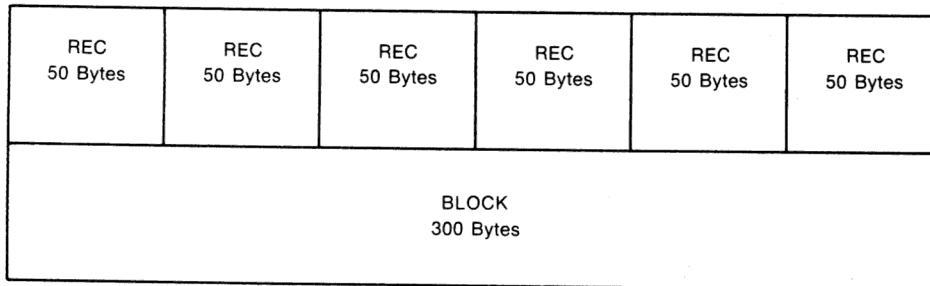
Fixed-length records are all the same length. No indication of the record length is required within the records because either the HDR2 label defines the record length, or you specify the record length with the /RECORDSIZE qualifier.

A fixed-length record can be a complete block, or several records can be grouped together in a block.

When records are blocked, it is not necessary that the block be filled with data. If a block is not filled, it will be padded with circumflex characters (^). This means that you cannot have a fixed-length record containing only circumflexes; the system will interpret this as padding, not data.

Figure B-6 shows a block of fixed-length records. Each record has a fixed length of 50 bytes. All six records are contained in a 300-byte block. The records are blocked—that is, grouped together as one entity—to increase processing efficiency; when records are blocked, you can access many of them with one I/O request. The block size should be a multiple of the record size.

**Figure B-6 Blocked Fixed-Length Records**

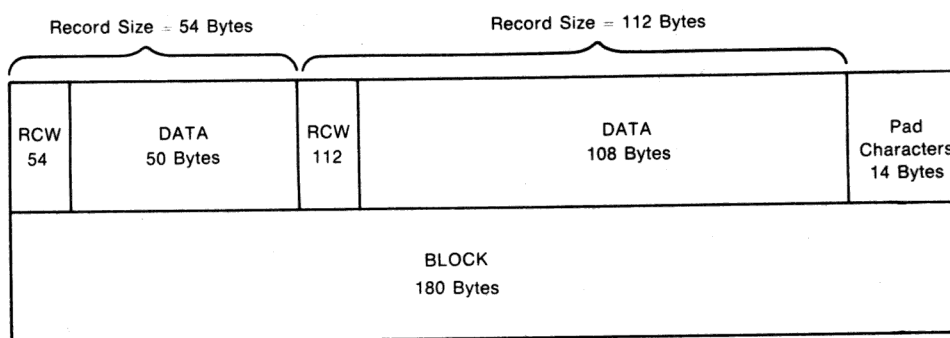


ZK-353-B1

The size of a variable-length record is indicated by a record control word (RCW). The RCW consists of four bytes at the beginning of each record. A variable-length record can be a complete block, or several records can be grouped together in a block.

Two variable-length records are shown in Figure B-7. The first consists of 54 bytes, including the RCW. The second consists of 112 bytes, including the RCW. The records are contained in a 166-byte block.

Do not use system-dependent record formats on volumes used for information interchange. VAX/VMS system-dependent formats are stream and variable with fixed-length control (VFC).

**Figure B-7 Variable-Length Records**

ZK-354-81

**B.6.2.2 Block Length Field**

The block length field denotes the maximum size of the blocks. According to the ANSI standard, valid block sizes range from 18 to 2048 bytes. However, VAX/VMS allows you to specify a smaller or larger block size using the /BLOCKSIZE qualifier with the MOUNT command. To specify the block size using RMS, see the BLS field in the file access block (FAB) in the *VAX Record Management Services Reference Manual*. When you specify a block size that is outside the ANSI standard range, the volume may not be processed correctly by other systems that support the ANSI standards.

**B.6.2.3 Record Length Field**

The record length field denotes either the size of fixed-length records or the maximum size of variable-length records in a file. Valid VAX/VMS record sizes vary, depending on the record format. The range for fixed-length records is 1 to 65,534 bytes; the range for variable-length records is 4 to 9,999 bytes, including the 4-byte RCW. Therefore, the maximum length of the data area of a variable-length record is 9,995 bytes. To comply with ANSI standards, the record size should not be larger than the maximum block size of 2,048 bytes, even though VAX/VMS allows larger record sizes (when the block size is larger).

For volumes containing files that do not have HDR2 labels, you must specify MOUNT/RECORDSIZE=n at mount time. The variable n denotes the record length in bytes. Files without HDR2 labels were created by a system that supports only level 1 or 2 of the ANSI standard for magnetic tape labels and file structure. Records should be fixed length because this is the only record format that either level supports. If you do not specify a record size, each block will be considered a single record.

---

#### B.6.2.4 Implementation-Dependent Field

The implementation-dependent field contains two 1-character subfields that describe how VAX/VMS interprets record format and form control.

The first subfield, character position 16, denotes whether the RMS attributes are in this label or the HDR3 label. If character position 16 contains a space, the RMS attributes are in the HDR3 label; if it contains any character other than a space, character position 16 is the first byte of the RMS attributes in the HDR2 label. The attributes appear in character positions 16 through 36 and 38 through 50.

For volumes created by any version of VAX/VMS up to and including Version 2.0, the system-dependent fields of the HDR2 label contain the RMS file attributes in binary format. For volumes created on versions of VAX/VMS later than Version 2.0 (Version 2.1 to the present), the magnetic tape file system stores the RMS attributes.

The second subfield, the form control field at character position 37, specifies the form control that defines the carriage control applied to records within a file. Possible values supported for VAX/VMS magnetic tape volumes are listed below.

- |       |  |
|-------|--|
| A     | First byte of record contains FORTRAN control characters.  |
| M     | The record contains all form control information.  |
| space | Line-feed/carriage-return combination is to be inserted between records when the records are written to a carriage-control device, such as a line printer or terminal. If form control is not specified when a file is created, this is the default. |

---

#### B.6.2.5 Buffer-Offset Length Field

For implementations that support buffer offsets, the buffer-offset length field indicates the length of information that prefixes each data block. The magnetic tape file system supports the input of buffer offset, which means that the buffer offset length obtained from the HDR2 label (when reading the file) is used to determine the actual start of the data block. The magnetic tape file system does not support the writing of a buffer offset.

Note that if you open a file for append or update access, and the buffer-offset length is nonzero, the open operation will not succeed.

---

### B.6.3 HDR3 LABEL

The HDR3 label describes the RMS file attributes. For RMS operations, data in the HDR3 label supersedes data in the HDR2 label. To write a VAX standard tape use the current ANSI standard.

Although the HDR3 label usually exists for every file on a VAX/VMS ANSI-labeled volume, there are two situations in which this label will not be written. The first is when an empty dummy file is created during volume initialization; no HDR3 label is written because the empty file does not require RMS attributes. The second is when you specify MOUNT/NOHDR3 at mount time. You should use the /NOHDR3 qualifier when you create files on volumes that will be interchanged to systems that do not process HDR3 labels correctly.

The RMS attributes describe the record format of a file. These attributes are converted from 32 bytes of binary values to 64 bytes of ASCII representations of their hexadecimal equivalents for storage in the HDR3 label.

---

### B.6.4 HDR4 Label

The HDR4 label contains the remainder of a VAX/VMS file name that would not fit in the HDR1 file identifier field. To write a VAX standard tape use the current ANSI standard.

---

## B.7 Trailer Labels

VAX/VMS supports two sets of trailer labels: end-of-file (EOF) and end-of-volume (EOV). A trailer label is written for each header label.

EOF and EOV labels are identical to their file header label counterparts except that:

- The label identifier field (characters 1-3) contains EOF or EOV.
- The block count field (characters 55-60) in the EOF1 and EOV1 labels contains the number of data blocks in the file section.

The particular label that VAX/VMS writes depends on whether a file extends beyond a volume. If a file terminates within the limits of a volume, EOF labels are written to delimit the file (see Figure B-2). If a file extends across volume boundaries before terminating, EOV labels are written, indicating that the file continues on another volume (see Figure B-3).



---

# Index

---

## A

---

### Access

- append • 4-23
- CONTROL • 2-2, 2-9, 2-11, 2-12
- DELETE • 2-2, 2-9, 2-11, 2-12
- EXECUTE • 2-2, 2-9, 2-11, 2-12
- file attributes • 4-22
- READ • 2-2, 2-9, 2-11
- to file • B-15

- magnetic tape • 4-19

- to volume

- magnetic tape • 4-19

- types of • 4-24

- update • 4-23

- WRITE • 2-2, 2-9, 2-11

### Access category

- GROUP • 2-2

- OWNER • 2-2

- summary of • 2-2

- SYSTEM • 2-2

- WORLD • 2-2

### Access control list

- See ACL

- Accessibility field • 2-13, B-10, B-15

### ACL (access control list)

- defining with DCL • 2-4

- description • 2-4

- modifying characteristics • 4-10

- ACL-based protection • 2-4

- See also ACL

- ALLOCATE command • 3-25

- See also Allocation

- magnetic tape • 4-18, 5-3

### Allocation

- of disk volume • 4-9, 4-10

- of drive • 3-2

- disk • 3-2

- generic • 3-3, 3-4

- magnetic tape • 3-3

### Allocation (cont'd.)

- of magnetic tape volume • 4-18

### Analyze/RMS\_File Utility

- (ANALYZE/RMS\_FILE) • 5-8

- ANSI data • B-1

- ANSI file name • 4-21, B-12

- ANSI-labeled volume • B-1, B-4

- accessibility protection • 2-5

- copying files from • 5-3

- Append access • 4-23

- ASCII "a" characters • 5-4, B-1

- ASCII "a" character set • B-1, B-4

- percent sign • 4-22

- Asterisk (\*) • 4-21, 4-22

- Automatic volume labeling (AVL)

- See AVL

- Automatic volume recognition (AVR)

- See AVR

- AVL operator function • 7-5

- AVR operator function • 7-5

---

## B

---

### BACKUP

- See Backup Utility

- BACKUP.SYS • A-4

- Backup home block • A-2

- Backup index file header • A-2

- Backup log file • A-4

- Backup Utility (BACKUP) • 6-2

- backup media • 6-6

- Files—11 disk • 6-7, 6-16

- magnetic tape • 6-7, 6-13

- remote save set • 6-9

- rotating backup set • 6-9

- sequential disk • 6-8

- batch mode • 7-8

- command procedure • 6-30

- comparing files • 6-27

- See also Compare operation

## Index

### Backup Utility (BACKUP) (cont'd.)

- copy
  - See Copy operation
- full-volume save • 6-18
- initializing a magnetic tape • 6-13
- invoking • 6-6
- journal file • 6-28
- journal files • 6-28
- listing files • 6-26
  - See also List operation
- magnetic tape • 6-7
- operations
  - compare • 6-27
    - See also Compare operation
  - copy • 6-11 to 6-13
    - See also Copy operation
  - image • 6-12, 6-15, 6-18
  - list • 6-26
    - See also List operation
  - overview of • 6-3 to 6-5
  - physical • 6-15
  - restore • 6-23 to 6-26
    - See also Restore operation
  - save • 6-13 to 6-19
    - See also Save operation
  - volume-by-volume • 6-19
- operator assistance requests • 7-8
- restore
  - See Restore operation
- restoring from image • 6-25
- save
  - See Save operation
- save set
  - See also Save set
  - protection • 6-22
  - restoring from • 6-23
- selective backups • 6-20 to 6-22
  - creation date • 6-21
  - expiration date • 6-21
  - /EXCLUDE qualifier • 6-22
  - UIC • 6-21
  - wildcard character • 6-20
- sequential disk save set • 6-16
  - multivolume • 6-17
- /COMPARE qualifier • 6-27

### Backup Utility (BACKUP)

- /COMPARE qualifier (cont'd.)
  - See also Compare operation
- /EXCLUDE qualifier • 6-22
- /IMAGE qualifier • 6-12, 6-15, 6-18, 6-25
  - See also Image operation
- /INITIALIZE qualifier • 6-17
- /JOURNAL qualifier • 6-28
- /LIST qualifier • 6-26, 6-28
  - See also List operation
- /LOG qualifier
  - directing output with • 6-11
- /OWNER\_UIC qualifier • 6-21, 6-22
- /PHYSICAL qualifier • 6-15
- /PROTECTION qualifier • 6-22
- /REWIND qualifier • 6-7, 6-13
- wildcard character
  - selective backups • 6-20
- BADBLK.SYS • A-3
- Bad block file • A-3
- BADLOG.SYS • A-4
- Batch job
  - accessing devices • 3-27
- Beginning-of-tape marker
  - See BOT
- Binary data • B-1
- Bit map
  - index file • A-2
  - storage • A-2
- BITMAP.SYS • A-2
- Blocked record • B-15
- Block length field • B-17
- Bootstrap block • 1-6, A-2
- BOT (beginning-of-tape) • B-2
- BOT marker • B-4
- Buffer-offset length field • B-18

---

## C

- Close operation • 4-22, 4-24
- Command procedure
  - accessing foreign volumes • 4-24
  - backup operations • 6-30
  - magnetic tape restriction • 4-2

## Index

- Command procedure (cont'd.)
  - setting up disk volume • 3-27
  - setting up magnetic tape volume • 3-28
  - setting up volume • 3-26
  - using to copy files • 5-13
- Compare operation (BACKUP) • 6-5, 6-27
- CONTIN.SYS • A-3
- Continuation file • A-3
- Continuation volume mounting • 3-21
- CONTROL access
  - See Access
- Convert Utility (CONVERT)
  - and non-file-structured volumes • 5-8
- COPY command • 5-1, 5-2
  - ANSI-labeled volumes
    - copying from • 5-4
  - copying from magnetic tape • 5-3
- DCL
  - copying files with • 4-2
  - disk files • 5-2
  - magnetic tape • 4-18
    - copying to • 5-3
  - non-file-structured volumes • 5-6
  - /LOG qualifier • 5-7
- Copy operation (BACKUP) • 6-11 to 6-13
  - copying files • 6-4, 6-11
  - copying multiple files • 6-11
  - disk volume
    - image • 6-12
  - disk volume set • 6-13
  - entire directory tree • 6-12
  - full volumes and volume sets • 6-18
- Core image file • A-3
- CORIMG.SYS • A-3
- Corruption of data • 3-25
- CREATE/DIRECTORY command • 4-17
- CREATE command
  - magnetic tape • 4-23
- Creation date field • B-14
  - zero creation date • B-15

---

## D

- DCL commands
  - restrictions on • 4-1

- DEALLOCATE command • 3-25
  - magnetic tape • 4-19
- DELETE access
  - See Access
- Device
  - See also Mount verification
  - accessing in batch job • 3-27
  - magnetic tape
    - retrieving information • 4-7
  - mounting volumes • 7-3
  - offline • 7-11
  - write lock • 7-13
- Directory
  - creating • 4-17
- DIRECTORY command • 4-2, 4-21
  - magnetic tape • 4-4, 5-3
  - /ACL qualifier • 4-8
  - /FULL qualifier • 4-22
- Disk
  - allocation of • 4-9, 4-10
  - allocation of space on • 1-4
  - block
    - cluster • 1-2
    - description • 1-2
    - record • 1-5
  - concepts
    - basic • 1-2
  - copying files • 5-2
  - deallocating drives • 3-25
  - default format • 5-2
  - file
    - copying
      - See also COPY command
      - See also Copy operation
      - copying to magnetic tape • 4-18
  - file characteristics
    - modifying • 4-12
  - mounting • 3-10
  - structure
    - Files-11 • 1-6
  - volume protection
    - See Protection
  - volume set
    - See Volume set
- Disk quota • 4-9

## Index

Disk structure  
Files-11 • A-1  
DISMOUNT command • 3-23, 3-25  
magnetic tape • 4-19  
/NOUNLOAD qualifier • 3-23  
/UNIT qualifier • 3-23  
DOS-11  
volume • 5-11  
Double tape mark • B-2

---

## E

End-of-tape marker  
See EOT  
EOF label • B-4, B-19  
EOT (end-of-tape) • B-2  
EOT marker • B-4  
EOV label • B-4, B-19  
Exchange Utility (EXCHANGE) • 5-1, 5-10  
DCL level • 5-12  
DIRECTORY command • 5-11  
exiting from • 5-12  
invoking • 5-11  
MOUNT command • 5-11  
EXECUTE access  
See Access  
Expiration date field • 4-19, B-14

---

## F

FAB BLS field • B-17  
File  
BACKUP  
using to compare • 6-27  
See also Compare operation  
copying from magnetic tape • 5-3  
copying to magnetic tape • 4-18  
format  
nonstandard • 4-2  
identifier field • B-12  
reserved  
Files-11 disk • A-1  
volume configurations • B-4, B-6, B-7, B-8, B-9

File access  
See Access  
File header • 1-6  
description • 1-7  
Files-11 structure • A-2  
File header label  
See Header label  
File name  
ANSI • 4-21  
File protection  
See Protection  
Files-11 disk  
BACKUP • 6-7  
Exchange Utility (EXCHANGE) • 5-11  
save set • 6-16  
structure • A-1  
Level 1 • 5-2  
Level 2 • 5-2  
reserved files • A-1  
structure levels compared • A-4  
File section number field • B-13  
File sequence number field • B-13  
File-set identifier field • B-13  
File specification • B-12  
ANSI • 4-21  
File type field • B-12  
Fixed-length record • B-15  
Foreign volumes  
mounting • 3-10  
Format  
ANSI-labeled volume • B-1

---

## G

Generation number • 4-20, B-13, B-14  
Generation-version number • 4-20, B-14

---

## H

HDR1 label  
See Header label

## Index

HDR2 label  
    See Header label  
HDR3 label  
    See Header label  
HDR label  
    See Header label  
Header label • B-4  
    HDR1 label • 4-20, B-11  
        accessibility field • B-15  
        creation date field • B-14  
        expiration date field • B-14  
        file identifier field • B-12  
        file section number field • B-13  
        file-set identifier field • B-13  
        generation number field • B-14  
        generation-version number field • B-14  
    HDR2 label • B-11, B-15  
        block length field • B-17  
        buffer-offset length field • B-18  
        record format field • B-15  
        record length field • B-17  
        system-dependent field • B-18  
    HDR3 label • B-11, B-19  
        RMS attributes field • B-19  
    HDR4 label • B-11, B-19  
Home block • 1-6, A-2

---

## I

Image operation • 6-18  
    copy • 6-12, 6-18  
    overview of • 6-3  
    restore • 6-18, 6-25  
    save • 6-15, 6-18  
INDEXF.SYS • A-2  
Index file • A-2  
    bit map • A-2  
    description • 1-6  
INITIALIZE command • 3-4  
    See also Volume  
    continuation volumes • 3-22  
    Files—11 On-Disk Structure • 3-6  
    magnetic tape • 5-3  
    protection codes • 4-13

INITIALIZE command (cont'd.)  
    sequential disk • 6-18  
Interchange environment  
    protection • 2-8

---

## J

Journal file • 6-28  
    See also Backup Utility

---

## L

Label  
    ANSI • B-1, B-4  
    EOF • B-19  
    EOV • B-19  
    HDR1 • B-11  
    HDR1 • 4-20  
    HDR2 • B-15  
    HDR3 • B-19  
    header • B-11  
    trailer • B-19  
    VOL1 • B-9  
List operation (BACKUP)  
    listing files • 6-5  
    wildcard characters • 6-27  
Logical name  
    MOUNT command • 3-8

---

## M

Magnetic tape  
    allocation of • 4-18  
    ANSI-labeled  
        mounting • 3-15  
    BACKUP  
        using with • 6-7  
    basic concepts of • 1-7  
    block • 1-7  
    copying files from • 5-3  
    deallocating drives • 3-25  
    density • 1-7  
    DOS—11 • 5-11  
    file • 1-9

## Index

- Magnetic tape
  - file (cont'd.)
    - reading • 4-22
  - file protection
    - See Protection
  - initializing • 3-6
  - initializing with BACKUP • 6-13
  - interrecord gap (IRG) • 1-7
  - modifying device characteristics • 4-12
  - mounting • 3-15
    - multiple volumes • 6-13
  - reading from • 4-22
  - record blocking • 1-8
  - record format • 5-3
  - retrieving device information • 4-7
  - runaway • 3-7
  - save set
    - multivolume • 6-13
    - restoring from • 6-23
    - writing to • 6-7, 6-14
  - specifying block size • 3-16
  - specifying record size • 3-19
  - 9-track drive • B-1
  - volume • 5-3
    - See Volume
  - volume protection
    - See Protection
  - volume set
    - See Volume set
  - write ring • 7-3
  - writing files to • 4-18, 4-23
- Magnetic tape ancillary control process
  - See MTAACP
- Master file directory
  - See MFD
- Media
  - See Backup Utility
  - See Disk
  - See Magnetic tape
- MFD (master file directory) • A-3
- sequential disk save sets • 6-8
- MOUNT command • 3-8, 3-25
  - See also ALLOCATE command
- MOUNT command (cont'd.)
  - See also DISMOUNT command
  - See also INITIALIZE command
  - See also Mounting
  - logical names • 3-8
  - magnetic tape • 5-3
  - protection codes • 4-13
  - qualifiers • 3-16
    - /ASSIST qualifier • 3-9
    - /AUTOMATIC qualifier • 3-22
    - /BIND qualifier • 3-11
    - /BLOCKSIZE qualifier • 3-16, 5-8, B-17
    - /CACHE=TAPE\_DATA qualifier • 3-18
    - /COMMENT qualifier
      - example • 7-3
    - /FOREIGN qualifier • 4-16, 5-6, 6-7, 6-8, 6-12, 6-13, 6-17, 6-24
    - /GROUP qualifier • 3-9
    - /HDR3 qualifier • 3-19
    - /INITIALIZE qualifier • 3-22
    - /LABEL qualifier • 3-16
    - /NOLABEL qualifier • 5-8
    - /OVERRIDE qualifier • 3-16, 4-19
    - /OWNER\_UIC qualifier • 3-17
    - /PROTECTION qualifier • 3-18
    - /RECORDSIZE qualifier • 3-19, 5-8, B-18
    - /SYSTEM qualifier • 3-9
  - specifying record size • 3-19
  - specifying UIC • 3-17
- Mount request • 3-9
  - MTAACP process • 7-5
- Mount verification
  - abort by dismount • 7-14
  - cancellation of • 7-14
  - operator functions • 7-11
  - procedure for device offline • 7-11
- MTAACP process • B-1
  - mount request • 7-5
- Multifile/multivolume configuration • B-9
- Multifile/single volume configuration • B-8
- Multivolume save set
  - magnetic tape • 6-13
- Multivolume sequential disk save set • 6-8

## O

### OPCOM

- mount verification • 7-11
- for offline devices • 7-11

### OPCOM message

- continuation volume request • 7-7
- device write-locked • 7-13
- enabling an operator terminal • 7-1
- mount request • 7-4
- mount verification aborted • 7-14
- mount verification completed • 7-12
- request display • 7-1
- unavailable device • 7-11
- wrong device • 7-12

### Operator Communication Facility

See OPCOM

### Operator function • 7-1 to 7-15

- mounting volume sets
  - with automatic switching • 7-5
  - without automatic switching • 7-6
- mount verification
  - canceling • 7-14
  - for write-locked devices • 7-13
  - handling tasks • 7-11
- user requests
  - for generic mount • 7-4
  - for mounting volumes • 7-3, 7-4
  - for mounting volume sets • 7-5
  - handling • 7-1
  - responding to • 7-2

### Operator terminal

- setting up • 7-1
- user requests • 7-1

## P

Pending bad block log file • A-4

Percent sign (%) • 4-22

Private volume

See Volume

Protection

- access category • 2-2

Protection (cont'd.)

- ACL-based • 2-4
- BACKUP save set • 6-22
- categories • 2-1
- code • 3-19
  - changing • 2-9
  - specifying • 2-9
- default
  - changing • 2-10
- file • 2-1
  - default • 2-10
  - directory • 2-8, 2-11
  - disk • 2-8, 2-9
  - magnetic tape • 2-3, 2-8, 2-13
- for interchange environments • 2-8
- UIC-based • 2-2
- volume • 2-1
  - ANSI-labeled • 2-5
  - disk • 2-6
  - magnetic tape • 2-6, 2-7

## Q

\$QIO call • B-1

## R

RCW (record control word) • B-16

READ access

See Access

Read operation • 4-16

continuation volumes • 7-6

disk • 4-16

magnetic tape • 4-19, 4-22

ANSI-labeled • 4-20

Record control word

See RCW

Record format • B-15

Record format field • B-15

Record length field • B-17

REPLY command

/BLANK\_TAPE qualifier • 7-6

/DISABLE qualifier • 7-2

/ENABLE qualifier • 7-1

/INITIALIZE\_TAPE qualifier • 7-6

/TO qualifier • 7-6

## Index

### Request

See Operator function

### REQUEST command

/REPLY qualifier • 7-1

/TO qualifier • 7-1

### Restore operation (BACKUP) • 6-5, 6-23 to 6-26

#### restoring files

from magnetic tape save sets • 6-23

from multivolume save sets • 6-25

from sequential disk save sets • 6-24

#### restoring volumes

entire disk volumes • 6-25

### RMS (Record Management System)

See VAX RMS

### Rotating backup set • 6-9

### RT-11

volume • 3-16

block-addressable • 5-11

---

## S

### Save operation (BACKUP) • 6-13 to 6-19

#### directories

saving to disks • 6-16

saving to magnetic tape • 6-13, 6-14

#### directory tree

saving to magnetic tape • 6-15

#### disk volume

saving an unstructured • 6-15

#### files

saving to disks • 6-16

saving to magnetic tape • 6-13, 6-14

#### file-structured disks

saving to • 6-16

#### full volumes and volume sets

saving • 6-18

#### image • 6-15

#### multiple volumes

to magnetic tape volumes • 6-15

#### saving files • 6-5

#### saving volumes

to magnetic tape • 6-13

#### sequential disk

multivolume, saving to • 6-17

### Save operation (BACKUP)

#### sequential disk (cont'd.)

saving to • 6-16

#### volume

saving to magnetic tape • 6-15

#### volumes

saving to disks • 6-16

### Save set • 6-6

comparing with disk files • 6-27

See also Compare operation

Files-11 disk • 6-7, 6-16

Files-11 disks • 6-16

listing contents of • 6-5, 6-26

See also List operation

magnetic tape • 6-7, 6-13, 6-14

multivolume • 6-13

name restriction • 6-7

protection • 6-22

remote • 6-9

restoring from • 6-23

sequential disk • 6-8, 6-16

multivolume • 6-8, 6-17

### Selective backup • 6-20 to 6-22

creation date • 6-21

expiration date • 6-21

/EXCLUDE qualifier • 6-22

#### UIC

using • 6-21

wildcard character • 6-20

### Sequential disk

save set on • 6-8

multivolume • 6-8

writing • 6-16, 6-17

### SET ACL command • 4-10

/ACL qualifier • 4-11

/LIKE qualifier • 4-11

/OBJECT\_TYPE qualifier • 4-10

### SET DIRECTORY command • 4-10, 4-11

/ACL qualifier • 4-11

/CONFIRM qualifier • 4-11

/OWNER\_UIC qualifier • 4-11

/VERSION\_LIMIT qualifier • 4-11

### SET FILE command • 4-10, 4-12

/ACL qualifier • 4-11

/BEFORE qualifier • 4-12

## Index

SET FILE command (cont'd.)  
    /EXPIRATION qualifier • 4-12  
    /OWNER\_UIC qualifier • 4-12  
SET MAGTAPE command • 4-10, 4-12  
SET PROTECTION command • 2-9, 4-8,  
    4-10, 4-14  
    /DEFAULT qualifier • 2-10, 4-13  
SET VOLUME command • 4-10  
    /DATA\_CHECK qualifier • 4-15  
    /LABEL qualifier • 4-15, 6-17  
SHOW ACL command • 4-2, 4-8  
SHOW command • 4-1  
SHOW DEVICE command • 3-21, 4-2, 4-4  
SHOW MAGTAPE command • 4-2, 4-7  
SHOW PROTECTION command • 4-2, 4-8  
SHOW QUOTA command • 4-2, 4-9  
Single file/multivolume configuration • B-7  
Single file/single volume configuration • B-6  
Storage bit map file • A-2  
Stream record type • B-16  
SUBMIT command • 4-2  
System-dependent field • B-18

## T

### Tape

    See Magnetic tape  
Tape mark • B-2, B-4  
Trailer label • B-4, B-19  
TYPE command  
    foreign volumes • 5-10  
    magnetic tape • 4-22

## U

UIC (user identification code)  
    backups  
        in selective • 6-21  
UIC-based protection • 2-2  
    to bypass • 2-3  
Unstructured disk volume  
    saving to magnetic tape with BACKUP •  
        6-15  
Update access • 4-23

User identification code  
    See UIC  
User requests  
    See Operator functions

## V

Variable-length record • B-16  
VAX RMS (Record Management Services) •  
    B-1, B-14, B-17  
    attributes • B-18, B-19  
Version number • 4-20, B-13  
VFC (variable with fixed-length control)  
    record format • B-16  
VOL1 label  
    See Volume label  
VOL label  
    See Volume label  
VOLSET.SYS • A-3  
Volume  
    ANSI-labeled • B-4  
        copying files from • 5-3  
    ANSI-labeled magnetic tape  
        mounting • 3-15  
    backing up full volumes and volume sets •  
        6-18  
    continuation • 3-22  
    file configurations • B-4, B-6, B-7, B-8, B-9  
    foreign • 3-10  
    initializing • 3-4, 3-6  
    label • B-4  
    magnetic tape • 5-3  
        copying files from • 5-3  
        deallocating • 4-19  
        dismounting • 4-19  
        initializing • 4-18  
        mounting • 3-15  
        record format • 5-3  
        writing files to • 4-18  
    modifying characteristics of disk • 4-15  
    mounting • 3-8, 3-10, 7-3  
        operator assistance • 7-3  
        operator functions • 7-3

## Index

### Volume (cont'd.)

- mounting with EXCHANGE • 5-11
- mounting without HDR2 labels • B-18
- operator assistance • 3-9
- owner field • B-11
- private • 3-1

### Volume identifier • B-10

### Volume label

- EOF label • B-4
- EOV label • B-4
- VOL1 label • B-9
  - accessibility field • B-10
  - volume identifier field • B-10
- VOL label • B-4

### Volume protection

- See Protection

### Volume RT-11 • 3-16

### Volume set

- creation of • 3-12, 3-13
- disk • 3-12
  - add volume • 3-14
  - mounting • 3-11
- initializing • 3-12
- list file • A-3
- loosely coupled • A-3
- magnetic tape
  - automatic volume switching • 3-21
  - continuation volumes • 3-21
  - creating • 3-20
  - mounting • 3-19
- mounting • 3-8, 3-11

---

## W

### Wildcard character (\*) • 4-3

- DIRECTORY command • 4-4
- file specification
  - magnetic tape • 4-20
- selective backup operations • 6-20

### WRITE access

- See Access

### Write lock

- mount verification • 7-13

### Write operation • 4-16

### Write operation (cont'd.)

- continuation volumes • 7-6
- disk • 4-17
- magnetic tape • 4-18, 4-19, 4-23
  - ANSI-labeled • 4-20

---

## X

### XABDAT block • B-14

- CDT field • B-14
- EDT field • B-14

---

## Z

### Zero creation date • B-15

## READER'S COMMENTS

**Note:** This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

---

---

---

---

---

Did you find errors in this manual? If so, specify the error and the page number.

---

---

---

---

---

Please indicate the type of user/reader that you most nearly represent:

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) \_\_\_\_\_

Name \_\_\_\_\_ Date \_\_\_\_\_

Organization \_\_\_\_\_

Street \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip Code \_\_\_\_\_  
or Country

Do Not Tear - Fold Here and Tape

**digital**



No Postage  
Necessary  
if Mailed in the  
United States

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SSG PUBLICATIONS ZK1-3/J35  
DIGITAL EQUIPMENT CORPORATION  
110 SPIT BROOK ROAD  
NASHUA, NEW HAMPSHIRE 03062-2698



Do Not Tear - Fold Here

Cut Along Dotted Line

## READER'S COMMENTS

**Note:** This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

---

---

---

---

---

---

Did you find errors in this manual? If so, specify the error and the page number.

---

---

---

---

---

---

Please indicate the type of user/reader that you most nearly represent:

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) \_\_\_\_\_

Name \_\_\_\_\_ Date \_\_\_\_\_

Organization \_\_\_\_\_

Street \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip Code \_\_\_\_\_  
or Country

Do Not Tear - Fold Here and Tape

**digital**



No Postage  
Necessary  
if Mailed in the  
United States

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SSG PUBLICATIONS ZK1-3/J35  
DIGITAL EQUIPMENT CORPORATION  
110 SPIT BROOK ROAD  
NASHUA, NEW HAMPSHIRE 03062-2698



Do Not Tear - Fold Here

Cut Along Dotted Line



